
PyGAF

Release 0.1

Tony Smith

Apr 16, 2023

CONTENTS

1	Contents	3
1.1	How to install and use PyGAF	3
1.2	PyGAF modules	3
1.3	Solution classes	30
	Python Module Index	79
	Index	81

PyGAF is a Python package for evaluation and display of analytic solutions for groundwater flow. The package name is an acronym of Python Groundwater Analytic Fflow.

Note: This project is under active development.

CONTENTS

1.1 How to install and use PyGAF

Install the development version of PyGAF from PyPI Test using pip

```
pip install -i https://test.pypi.org/simple/ pygaf==0.1.0
```

Jupyter notebooks demonstrating the use of PyGAF are available at GitHub

https://github.com/Moofbytes/pygaf_notebooks

1.2 PyGAF modules

1.2.1 Aquifer classes

```
class pygaf.aquifers.Aq1dFiniteConf(K=1, Ss=0.0001, B=10, L=1000, bot=0, name='Aq1dFiniteConf'  
                                     class)
```

Bases: *Aquifer*

1D, finite confined aquifer subclass.

A subclass of the *Aquifer* class defining a horizontal, 1D confined aquifer with finite lateral extent and confined storage.

The default *Aq1dFiniteConf* object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer length $L=1000$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss , B or L .

The `.info` and `.draw` methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units $1/L$, default $1.0e-4$).

Type

float

L

Aquifer length (units L , default 1000.0).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type
float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property L

Aquifer length (units L, default 1000.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type
float

property Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

draw(dw=6)

Display a drawing of the aquifer.

Parameters
dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = True

is_2d = False

is_confined = True

is_finite = True

is_heterogeneous = False


```

is_homogeneous = True
is_infinite = False
is_leaky = False
is_radial = False
is_semifinite = False
is_unconfined = False

```

```

class pygaf.aquifers.Aq1dFiniteUnconf(K=1, Sy=0.1, B=10, L=1000, bot=0, name='Aq1dFiniteUnconf
                                     class')

```

Bases: [Aquifer](#)

1D, finite unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 1D unconfined aquifer with finite lateral extent and unconfined storage.

The default Aq1dFiniteUnconf object has hydraulic conductivity $K=1$, specific yield $Sy=0.1$, static saturated thickness $B=10$, aquifer length $L=1000$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Sy , B or L .

The `.info` and `.draw` methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type
float

L

Aquifer length (units L, default 1000.0).

Type
float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property L

Aquifer length (units L, default 1000.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property T

Aquifer transmissivity (units L²/T, default 10.0).

Type

float

draw(*dw=6*)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = True

is_2d = False

is_confined = False

is_finite = True

is_heterogeneous = False

is_homogeneous = True

is_infinite = False

is_leaky = False

is_radial = False

is_semifinite = False

is_unconfined = True

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

```
class pygaf.aquifers.Aq1dSemifiniteConf(K=1, Ss=0.0001, B=10, bot=0, name='Aq1dSemifiniteConf'  
                                         class)
```

Bases: *Aquifer*

1D, semi-infinite confined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 1D confined aquifer with semi-finite (semi-infinite) lateral extent and confined storage.

The default Aq1dSemifiniteConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type
float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property D

Aquifer diffusivity (units L^2/T , default 1.0e+4).

Type
float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type
float

property Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

draw(dw=6)

Display a drawing of the aquifer.

Parameters
dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = True

is_2d = False

is_confined = True

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = False

is_leaky = False

is_radial = False

is_semifinite = True

is_unconfined = False

class pygaf.aquifers.**Aq1dSemifiniteUnconf**(*K=1, Sy=0.1, B=10, bot=0, name='Aq1dSemifiniteUnconf class'*)

Bases: *Aquifer*

1D, semi-infinite unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 1D unconfined aquifer with semi-finite (semi-infinte) lateral extent and unconfined storage.

The default Aq1dFiniteUnconf object has hydraulic conductivity $K=1$, specific yield $Sy=0.1$, static saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Sy or B .

The .info and .draw methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property T

Aquifer transmissivity (units L²/T, default 10.0).

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = True

is_2d = False

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = False

is_leaky = False

is_radial = False

is_semifinite = True

is_unconfined = True

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

class pygaf.aquifers.**Aq2dConf**(*K=1, Ss=0.0001, B=10, bot=0, name='Aq2dConf class'*)

Bases: *Aquifer*

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L^2/T , default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*dw=6*)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = True

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = False

class pygaf.aquifers.**Aq2dLeaky**(*K=1, Ss=0.0001, B=10, bot=0, Kleak=1e-05, Bleak=10, name='Aq2dLeaky class'*)

Bases: [*Aquifer*](#)

2D leaky aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D leaky aquifer with infinite lateral extent and confined storage.

The default Aq2dLeak object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer bottom (datum) elevation $bot=0$, leaky hydraulic conductivity $Kleak=0.00001$ and leaky thickness $Bleak=10$. Exceptions will occur if invalid values are provided for K , Ss , B , $Kleak$ or $Bleak$.

The `.info` and `.draw` methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Bleak

Leaky layer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L^2/T , default $1.0e+4$).

Type

float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Kleak

Leaky layer hydraulic conductivity (units L/T , default 0.00001).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default $1.0e-3$).

Type

float

property Ss

aquifer specific storativity (units 1/L, default $1.0e-4$).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*dw=6*)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = True


```
is_semifinite = False
```

```
is_unconfined = False
```

```
class pygaf.aquifers.Aq2dUnconf(K=1, Sy=0.1, B=10, bot=0, name='Aq2dUnconf class')
```

Bases: [Aquifer](#)

2D unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D unconfined aquifer with infinite lateral extent and unconfined storage.

The default Aq2dUnconf object has hydraulic conductivity $K=1$, specific yield $Sy=0.1$, static saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Sy or B .

The `.info` and `.draw` methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

```
is_1d = False
is_2d = True
is_confined = False
is_finite = False
is_heterogeneous = False
is_homogeneous = True
is_infinite = True
is_leaky = False
is_semifinite = False
is_unconfined = True
```

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

```
class pygaf.aquifers.Aquifer(K=1, B=10, bot=0, name='Parent aquifer class')
```

Bases: object

Aquifer parent class.

The Aquifer parent class defines attributes and properties common to all aquifers. They include the aquifer transmissivity (defined by hydraulic conductivity and saturated thickness), the aquifer elevation datum and an aquifer name label for use in figures.

K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Type

float

B

Aquifer thickness (units L, default 10.0).

Type

float

bot

Aquifer bottom elevation (units L reduced level, default 0.0).

Type

float

name

Aquifer label (default 'Unnamed').

Type

str

property T

Aquifer transmissivity (units L²/T, default 10.0).

Type

float

property top

Aquifer top elevation (units L, default 10.0).

Type

float

1.2.2 Basin classes

class pygaf.basins.CircBasin(*cx=0.0, cy=0.0, diam=10, name='Circle basin'*)

Bases: object

Circular basin.

cx

Basin center x coordinate (units L, default 0.0).

Type

float

cy

Basin center y coordinate (units L, default 0.0).

Type

float

diam

Basin diameter (default 10.0).

Type

float

rot

Basin rotation angle in radians (default 0.0).

Type

float

name

Basin name (default 'Unnamed').

Type

str

property area

basin area.

Type

float

property diam

Basin diameter.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*dw=4*)

Display a drawing of the basin.

Parameters**dw** (*float*) – Width of basin drawing (default 4.0).**info**()

Print the basin information.

is_circular = **True****is_rectangular** = **False****property** **rad**

Basin radius.

Type

float

class pygaf.basins.**RectBasin**(*cx=0.0, cy=0.0, lx=10, ly=10, rot=0, name='Rectangle basin'*)

Bases: object

Rectangular basin.

cx

Basin center x coordinate (units L, default 0.0).

Type

float

cy

Basin center y coordinate (units L, default 0.0).

Type

float

lx

Basin length in x direction (default 10.0)

Type

float

ly

Basin length in y direction (default 10.0)

Type

float

rot

Basin rotation angle in radians (default 0.0).

Type

float

name

Basin name (default 'Unnamed').

Type

str

property area

Basin area.

Type

float

draw(dw=4)

Display a drawing of the basin.

Parameters

dw (*float*) – Width of basin drawing (default 4.0).

info()

Print the basin information.

is_circular = False**is_rectangular = True****property lx**

Basin length in x direction.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property ly

Basin length in y direction.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot

Basin rotation angle in degrees.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot_rad

Basin rotation angle in radians.

Type

float

property verts

x and y coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type

dict

property verts_rot

x and y rotated coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type

dict

1.2.3 Steady state boundary conditions

class pygaf.bcs.SteadyBC(*type=2, head=10.0, flow=0.0, cond=0.0*)

Bases: object

Boundary condition for steady state groundwater flow.

The boundary condition can be type 1, 2 or 3 (see the 'type' attribute). An exception occurs if any other value is specified for type. The default SteadyBC object is a type 2 boundary condition with flow=0.0, which is equivalent to a no-flow boundary. The default type 1 Boundary has head=10.0 (same as the default head of all Aquifer classes). The default type 3 boundary condition has head=10.0 and cond=0.0, which is equivalent to a no-flow condition.

Note a utilities function is available to calculate conductance values

type

Boundary condition type; choices are type=1 (Dirichlet, first-type or constant head), type=2 (Neumann, second-type or constant flow) or type=3 (Cauchy or general head).

Type

int

head

Value of head at the boundary for type 1 and value of

Type

float

external head for type 2

Type

units L, default 10.0

flow

value of normal flow at boundary for type 2 (units L/T, default 0.0).

Type

float

cond

Value of conductance for type 3 (units L²/T, default 0.0).

Type

float

info()

Print the solution information.

property type

Boundary condition type.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

int

property value

Boundary condition value(s).

Type

dic

1.2.4 Grid classes

class pygaf.grids.**BasinGrid**(*gr=100, gd=21*)

Bases: object

Square grid class with basin center at grid center.

The default BasinGrid object has radius *gr*=100 defining the square's extent and grid density *gd*=21. An exception occurs if the grid radius is not positive. Grid density defines the numbers of rows and columns comprising the grid; thus, the default grid has 21 rows x 21 cols = 441 grid points. Minimum grid density is constrained to 11 (121 grid points) and maximum grid density is constrained to 41 (1681 grid points). Values for *gd* outside of these constraints are re-set to the minimum or maximum values as appropriate.

The *.pts* property returns the grid points attributes including local x-y coordinates and world x-y coordinates. The *.info* method displays grid information and the *.draw* method displays a plot of the grid in either local or world coordinates.

gr

Radius defining the extent of the solution grid (units L, default 100.0).

Type

float

gd

Grid density defining the number of rows and columns; minimum and maximum constraints are enforced (default 21).

Type

int

class RectBasin(*cx=0.0, cy=0.0, lx=10, ly=10, rot=0, name='Rectangle basin'*)

Bases: object

Rectangular basin.

cx

Basin center x coordinate (units L, default 0.0).

Type

float

cy

Basin center y coordinate (units L, default 0.0).

Type

float

lx

Basin length in x direction (default 10.0)

Type

float

ly

Basin length in y direction (default 10.0)

Type

float

rot

Basin rotation angle in radians (default 0.0).

Type

float

name

Basin name (default 'Unnamed').

Type

str

property area

Basin area.

Type

float

draw(dw=4)

Display a drawing of the basin.

Parameters

dw (*float*) – Width of basin drawing (default 4.0).

info()

Print the basin information.

is_circular = False**is_rectangular = True****property lx**

Basin length in x direction.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property ly

Basin length in y direction.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot

Basin rotation angle in degrees.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot_rad

Basin rotation angle in radians.

Type

float

property verts

x and y coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type

dict

property verts_rot

x and y rotated coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type
dict

draw(*local=False*)

Draw the grid points.

Parameters

local (*bool*) – Display the grid plot in local coordinates with the well at 0, 0 (default False).

property gr

Grid radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property grdim

Number of grid rows and columns.

Type
int

info()

Print the basin grid information.

property npts

Number of grid points.

Type
int

property pts

grid point attributes including local grid point coordinates and world grid point coordinates.

Type
pandas dataframe

class pygaf.grids.**WellGrid**(*gr=100, gd=21*)

Bases: object

Square grid class with regular spacing and well at grid center.

The default WellGrid object has radius *gr*=100 defining the square's extent and grid density *gd*=21. An exception occurs if the grid radius is not positive. Grid density defines the numbers of rows and columns comprising the grid; thus, the default grid has 21 rows x 21 cols = 441 grid points. Minimum grid density is limited to 11 (121 grid points) and maximum grid density is limited to 41 (1681 grid points). Values for *gd* outside of these constraints are re-set to the minimum or maximum values as appropriate.

The *.pts* property returns the grid points attributes including local x-y coordinates, world x-y coordinates and radius values relative to the well location. The *.info* method displays grid information and the *.draw* method displays a plot of the grid in either local or world coordinates.

gr

Radius defining the extent of the solution grid (units L, default 100.0).

Type
float

gd

Grid density defining the number of rows and columns; minimum and maximum constraints are enforced (default 21).

Type

int

class Well(*x=0.0, y=0.0, r=0.05, pf=1.0, name='Steady state non-flowing well'*)

Bases: **object**

Non-flowing well.

x

Well x coordinate (units L, default 0.0).

Type

float

y

Well y coordinate (units L, default 0.0).

Type

float

r

Well radius (units L, default 0.05).

Type

float

pf

Well penetration depth (fraction of aquifer depth, default 1.0).

Type

float

name

Well name (default ‘’).

Type

str

info()

Print the well information.

property pf

Well penetration depth.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property r

Well radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*local=False*)

Draw the grid points.

Parameters

local (*bool*) – Display the grid plot in local coordinates with the well at 0, 0 (default False).

property gr

Grid radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property grdim

Number of grid rows and columns.

Type

int

info()

Print the well grid information.

property npts

Number of grid points.

Type

int

property pts

grid point attributes including local grid point coordinates, world grid point coordinates and radius values of grid points relative to the well center.

Type

pandas dataframe

1.2.5 Stress classes

```
class pygaf.stresses.StressSeries(periods=[1.0], values=[0.0], from_csv="", title='Stress Series',
                                xlabel='Time', ylabel='Value')
```

Bases: object

Stress series class defining stress periods and values.

The default StressSeries object has default values `periods=[1.0]` and `values=[0.0]`. Stress periods and values are provided in corresponding lists, which can be created manually or loaded from a csv file containing comma-separated period and value pairs, one per line. Exceptions occur if a period is negative or if the number of periods and stresses do not match.

The `.plot` method displays a timeseries graph of the stress series.

Example csv file with three stress periods:

10,25.6

15.2,38.7

48,-12

periods

List of stress period lengths; used if `from_csv` is an empty string (units T, default [1.0]).

Type
float

values

List of stress period values, one per stress period; used if from_csv is an empty string (units consistent, default [0.0]).

Type
float

from_csv

File path of csv file to read stress data from; no data are read if the string is empty (default ‘’).

Type
str

property periods

List of stress period lengths.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
list (float)

plot(dw=8)

Plot the stress series.

Parameters
dw (*float*) – Width of figure (default 8).

property series

stress series dataframe.

Type
pandas dataframe

property values

List of stress period values.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
list (float)

1.2.6 Utilities

`pygaf.utils.add_constant_to_list(list, const)`

Add a constant value to each item of a list.

Parameters

- **list** (*float*) – 2d list of numeric values.
- **const** (*float*) – Constant value to add to each list item.

Returns

2d list.

`pygaf.utils.conductance(K=1.0, B=1.0, W=1.0, L=1.0)`

Calculate conductance value from components.

Parameters

- **K** (*float*) – Hydraulic conductivity of external porous medium (units L/T, default 1.0).
- **B** (*float*) – Thickness of external porous media normal to flow (units L, default 1.0).
- **W** (*float*) – Width of external porous medium normal to flow (units L, default 1.0); should be equal to 1.0 for 1D flow,
- **L** (*float*) – Length of external porous medium between the boundary and external head (units L, default 1.0).

`pygaf.utils.deg2rad(deg)`

Convert degrees to radians for an angle between -90 and 90 deg.

Parameters

deg (*float*) – Angle in degrees.

Returns

Angle in radians.

`pygaf.utils.display_image(fname, dw=8)`

Display an image file from the images folder.

Parameters

- **fname** (*str*) – Image file name.
- **dw** (*float*) – Width of figure (default 8.0).

`pygaf.utils.rotate_grid(x0, y0, x, y, phi)`

Rotate the coordinates of a grid.

Parameters

- **x0** (*float*) – X coordinate of center of rotation.
- **y0** (*float*) – Y coordinate of center of rotation.
- **x** (*float*) – 1d list of grid x coordinates.
- **y** (*float*) – 1d list of grid y coordinates.
- **phi** (*float*) – Angle of clockwise rotation in radians.

Returns

Rotated x as 1d list, rotated y as 1d list.

`pygaf.utils.rotate_grid_2d(x0, y0, x, y, phi)`

Rotate the coordinates of a grid in 2d format.

Parameters

- **x0** (*float*) – X coordinate of center of rotation.
- **y0** (*float*) – Y coordinate of center of rotation.
- **x** (*float*) – 2d list of grid x coordinates.
- **y** (*float*) – 2d list of grid y coordinates.
- **phi** (*float*) – Angle of clockwise rotation in radians.

Returns

Rotated x as 2d list, rotated y as 2d list.

`pygaf.utils.rotate_point(x0, y0, x1, y1, phi)`

Rotate the coordinates of a point.

Parameters

- **x0** (*float*) – X coordinate of ceter of rotation.
- **y0** (*float*) – Y coordinate of center of rotation.
- **x1** (*float*) – X coordinate of point to be rotated.
- **y1** (*float*) – Y coordinate of point to be rotated.
- **phi** (*float*) – Angle of clockwise rotation in radians.

Returns

Tupple of rotated x and y coordinates.

1.2.7 Well classes

class `pygaf.wells.SteadyWell`(*x=0.0, y=0.0, r=0.05, q=0.0, pf=1, name='Steady state flowing well'*)

Bases: object

Steady state flowing well.

The default SteadyWell object has coordinates $x=0.0$ and $y=0.0$, well radius $r=0.05$, penetration fraction $pf=1.0$ and well rate $q=0.0$. Exceptions occur if invalid values are provided for r or pf . The well rate can be negative, positive or zero and is used to set the `.state` attribute as extract, inject or off.

x

Well x coordinate (units L, default 0.0).

Type

float

y

Well y coordinate (units L, default 0.0).

Type

float

r

Well radius (units L, default 0.05).

Type

float

q

Well rate (units L^3/T , default 0.0).

Type

float

pf

Well penetration depth (fraction of aquifer depth, default 1.0).

Type

float

name

Well name (default '').

Type

str

info()

Print the well information.

is_steady = True**is_transient = False****property pf**

Well penetration depth.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property r

Well radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property state

Well state.

Type

str

class pygaf.wells.**TransientWell**(*x=0.0, y=0.0, r=0.05, pf=1, name='Transient flowing well'*)

Bases: object

Transient state flowing well.

The default TransientWell object has well coordinates $x=0.0$ and $y=0.0$, well radius $r=0.05$, penetration fraction $pf=1.0$ and default StressSeries object. Exceptions occur if invalid values are provided for r or pf . The well rate can be negative, positive or zero and is used to set the `.state` attribute as extract, inject or off.

x

Well x coordinate (units L, default 0.0).

Type

float

y

Well y coordinate (units L, default 0.0).

Type

float

r

Well radius (units L, default 0.05).

Type

float

ss

StressSeries object.

Type

obj

pf

Well penetration depth (fraction of aquifer depth, default 1.0).

Type

float

name

Well name (default '').

Type

str

```
class StressSeries(periods=[1.0], values=[0.0], from_csv="", title='Stress Series', xlabel='Time',  
                    ylabel='Value')
```

Bases: `object`

Stress series class defining stress periods and values.

The default StressSeries object has default values periods=[1.0] and values=[0.0]. Stress periods and values are provided in corresponding lists, which can be created manually or loaded from a csv file containing comma-separated period and value pairs, one per line. Exceptions occur if a period is negative or if the number of periods and stresses do not match.

The .plot method displays a timeseries graph of the stress series.

Example csv file with three stress periods:

10,25.6

15.2,38.7

48,-12

periods

List of stress period lengths; used if from_csv is an empty string (units T, default [1.0]).

Type

float

values

List of stress period values, one per stress period; used if from_csv is an empty string (units consistent, default [0.0]).

Type

float

from_csv

File path of csv file to read stress data from; no data are read if the string is empty (default '').

Type

str

property periods

List of stress period lengths.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

list (float)

plot(dw=8)

Plot the stress series.

Parameters

dw (*float*) – Width of figure (default 8).

property series

stress series dataframe.

Type

pandas dataframe

property values

List of stress period values.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

list (float)

info()

Print the well information.

is_steady = False

is_transient = True

pandas = <module 'pandas' from '/home/docs/checkouts/readthedocs.org/user_builds/pygaf/envs/latest/lib/python3.7/site-packages/pandas/__init__.py'>

property pf

Well penetration depth.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property r

Well radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property state

Well state.

Type

str

class pygaf.wells.**Well**(x=0.0, y=0.0, r=0.05, pf=1.0, name='Steady state non-flowing well')

Bases: object

Non-flowing well.

x

Well x coordinate (units L, default 0.0).

Type

float

y

Well y coordinate (units L, default 0.0).

Type

float

r

Well radius (units L, default 0.05).

Type

float

pf

Well penetration depth (fraction of aquifer depth, default 1.0).

Type

float

name

Well name (default '').

Type

str

info()

Print the well information.

property pf

Well penetration depth.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property r

Well radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

1.3 Solution classes

1.3.1 Glover (1960) recharge basin

class pygaf.solutions.glover_1960.GloverRectBasinSteady

Bases: object

Glover (1960) solution class.

The default GloverRectBasinSteady object uses the default Aq2dConf and BasinGrid classes. It's methods include impress at a point .impress and grid-contoured impress at specified time .impress_grid.

aq

Confined aquifer object.

Type
obj

basin

Basin object.

Type
obj

class Aq2dConf(*K=1, Ss=0.0001, B=10, bot=0, name='Aq2dConf class'*)

Bases: [Aquifer](#)

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default $1.0e-4$).

Type
float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property D

Aquifer diffusivity (units L^2/T , default $1.0e+4$).

Type
float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

property S

Aquifer storage coefficient (units 1, default $1.0e-3$).

Type
float

property Ss

aquifer specific storativity (units 1/L, default $1.0e-4$).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*dw=6*)

Display a drawing of the aquifer.

Parameters**dw** (*float*) – Width of figure (default 6.0).**info**()

Print the aquifer information.

is_1d = **False****is_2d** = **True****is_confined** = **True****is_finite** = **False****is_heterogeneous** = **False****is_homogeneous** = **True****is_infinite** = **True****is_leaky** = **False****is_semifinite** = **False****is_unconfined** = **False****class BasinGrid**(*gr=100, gd=21*)

Bases: object

Square grid class with basin center at grid center.

The default BasinGrid object has radius *gr*=100 defining the square's extent and grid density *gd*=21. An exception occurs if the grid radius is not positive. Grid density defines the numbers of rows and columns comprising the grid; thus, the default grid has 21 rows x 21 cols = 441 grid points. Minimum grid density is constrained to 11 (121 grid points) and maximum grid density is constrained to 41 (1681 grid points). Values for *gd* outside of these constraints are re-set to the minimum or maximum values as appropriate.

The *.pts* property returns the grid points attributes including local x-y coordinates and world x-y coordinates. The *.info* method displays grid information and the *.draw* method displays a plot of the grid in either local or world coordinates.

gr

Radius defining the extent of the solution grid (units L, default 100.0).

Type

float

gd

Grid density defining the number of rows and columns; minimum and maximum constraints are enforced (default 21).

Type

int

```

class RectBasin(cx=0.0, cy=0.0, lx=10, ly=10, rot=0, name='Rectangle basin')
    Bases: object
    Rectangular basin.

    cx
        Basin center x coordinate (units L, default 0.0).
        Type
        float

    cy
        Basin center y coordinate (units L, default 0.0).
        Type
        float

    lx
        Basin length in x direction (default 10.0)
        Type
        float

    ly
        Basin length in y direction (default 10.0)
        Type
        float

    rot
        Basin rotation angle in radians (default 0.0).
        Type
        float

    name
        Basin name (default 'Unnamed').
        Type
        str

    property area
        Basin area.
        Type
        float

    draw(dw=4)
        Display a drawing of the basin.
        Parameters
            dw (float) – Width of basin drawing (default 4.0).

    info()
        Print the basin information.

    is_circular = False

    is_rectangular = True

    property lx
        Basin length in x direction.
        Setter method checks for valid values and triggers an exception if invalid values are specified.
        Type
        float

```

property ly

Basin length in y direction.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot

Basin rotation angle in degrees.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property rot_rad

Basin rotation angle in radians.

Type

float

property verts

x and y coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type

dict

property verts_rot

x and y rotated coordinates of basin verticies.

Vertex keys: ll - lower left, ul - upper left, lr - lower right and ur - upper right.

Type

dict

draw(local=False)

Draw the grid points.

Parameters

local (*bool*) – Display the grid plot in local coordinates with the well at 0, 0 (default False).

property gr

Grid radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property grdim

Number of grid rows and columns.

Type

int

info()

Print the basin grid information.

property npts

Number of grid points.

Type

int

property pts

grid point attributes including local grid point coordinates and world grid point coordinates.

Type

pandas dataframe

h(*x*, *y*, *xL*, *yL*, *T*, *S*, *t*, *q*)

Glover impress solution.

impress(*t*=[1], *locs*=[(0, 0)], *q*=0.0, *plot*=True, *csv*="", *xlsx*="")

Calculate impress at specified locations and times.

Parameters

- **t** (*float*) – List of times to evaluate impress (default [1.0]).
- **locs** (*float tuple*) – List of (dx, dy) location tuples to evaluate impress; dx is distance from basin center in x direction; dy is distance from basin center in y direction (default [(0.0, 0.0)]).
- **q** (*float*) – Basin infiltration rate (default 0.0).
- **plot** (*bool*) – Display a plot of the results (default True).
- **csv** (*str*) – Filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Pandas dataframe containing results, hydraulic loading.

impress_grid(*t*=1, *q*=0.0, *gr*=100, *gd*=20, *plot*=True, *local*=False, *csv*="", *xlsx*="")

Calculate impress on a regular grid.

Parameters

- **t** (*float*) – Time to evaluate impress (default [1.0]).
- **q** (*float*) – Basin infiltration rate (default 0.0).
- **gr** (*float*) – Radius defining the extent of the solution grid (default 100.0).
- **gd** (*int*) – Grid density defining the number of grid rows and columns; minimum and maximum constraints are enforced (default 21).
- **plot** (*bool*) – Display a plot of the results (default True).
- **local** (*bool*) – Display the drawdown plot in 'local' coordinates with the well at 0, 0 (Default False).
- **csv** (*str*) – Filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Pandas dataframe containing results, hydraulic loading.

info()

Print the solution information.

Returns

Screen printout of solution information.

u1(*x, xL, T, S, t, tau*)

Glover u1 solution term; tau is an integration variable.

u2(*x, xL, T, S, t, tau*)

Glover u2 solution term; tau is an integration variable.

u3(*y, yL, T, S, t, tau*)

Glover u3 solution term; tau is an integration variable.

u4(*y, yL, T, S, t, tau*)

Glover u4 solution term; tau is an integration variable.

1.3.2 Mine pit inflow solutions

class `pygaf.solutions.mine_flow.MineSteadyRadLeakyDD`

Bases: `object`

Steady, radial, leaky flow to a large diameter well.

Predict steady drawdown for a specified mine pit inflow. The `MineSteadyRadleakyDD` class uses the default `Aq2dLeaky` aquifer object with aquifer thickness 100 for the lower confined aquifer.

aq

Lower aquifer object.

Type

`obj`

class `Aq2dLeaky(K=1, Ss=0.0001, B=10, bot=0, Kleak=1e-05, Bleak=10, name='Aq2dLeaky class')`

Bases: `Aquifer`

2D leaky aquifer subclass.

A subclass of the `Aquifer` class defining a horizontal, 2D leaky aquifer with infinite lateral extent and confined storage.

The default `Aq2dLeak` object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer bottom (datum) elevation $bot=0$, leaky hydraulic conductivity $Kleak=0.00001$ and leaky thickness $Bleak=10$. Exceptions will occur if invalid values are provided for K , Ss , B , $Kleak$ or $Bleak$.

The `.info` and `.draw` methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default $1.0e-4$).

Type

`float`

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

`float`

property Bleak

Leaky layer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Kleak

Leaky layer hydraulic conductivity (units L/T, default 0.00001).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = True

is_semifinite = False

is_unconfined = False

dd(*n=25, plot=True, csv="", xlsx=""*)

Evaluate distance-drawdown.

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting `plot=False`.

Parameters

- **n** (*int*) – Number of radial values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default “”).
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default “”).

Returns

Results in a pandas dataframe.

dr(*r*)

Drawdown at specified radius (units L).

Parameters

r (*float*) – radius at which to evaluate drawdown (units L).

draw(*dw=8*)

Display the definition diagram.

Parameters

dw (*float*) – Width of figure (default 8.0).

property h0

Initial groundwater head (units L, default 120).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hr(*r*)

Head at sepcified radius (units L).

Parameters

r (*float*) – radius at which to evaluate head (units L).

info()

Print the solution information.

leakr(*r*)

Leakage rate at specified radius (units L/T).

Parameters

r (*float*) – radius at which to evaluate leakage (units L).

property lfac

Aquitard leakage factor (units L).

Type

float

property qp

Mine pit inflow rate (units L/T, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property ri

Radius of influence (units L).

Defined as radius at which drawdown is less than 0.1% of initial groundwater head.

Type

float

property rp

Equivalent radius of mine pit (units L, default 100).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

class pygaf.solutions.mine_flow.MineSteadyRadUnconfQ

Bases: object

Steady, radial, unconfined flow to a large diameter well.

Predict steady mine inflow for a specified mine pit water level. The MineSteadyRadUnconfQ class uses the default Aq2dUnconf aquifer object with initial saturated thickness 100.

aq

Aquifer object.

Type

obj

class Aq2dUnconf(*K=1, Sy=0.1, B=10, bot=0, name='Aq2dUnconf class'*)

Bases: [Aquifer](#)

2D unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D unconfined aquifer with infinite lateral extent and unconfined storage.

The default Aq2dUnconf object has hydraulic conductivity $K=1$, specific yield $Sy=0.1$, static saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Sy or B .

The .info and .draw methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = True

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

property R

Groundwater recharge rate (units L/T, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

dd(*n=25, plot=True, csv="", xlsx=""*)

Evaluate distance-drawdown (units L).

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting `plot=False`.

Parameters

- **n** (*int*) – Number of radial values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default ‘’).
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default ‘’).

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level from initial water table (units L, default 10).

Type

float

dr(*r*)

Drawdown at specified radius (units L).

Parameters

r (*float*) – Radius at which to evaluate head (units L).

draw(*dw=8*)

Display the definition diagram.

Parameters

dw (*float*) – Width of figure (default 8.0).

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hr(*r*)

Head at specified radius (units L).

Parameters

r (*float*) – Radius at which to evaluate head (units L).

info()

Print the solution information.

property qp

Mine pit inflow rate (units L³/T).

Type

float

property ri

Radius of influence (units L).

Type

float

property rp

Equivalent radius of mine pit (units L, default 100).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

class pygaf.solutions.mine_flow.**MineSteadyRadUnconfQ2**

Bases: object

Steady, radial, unconfined flow to a large diameter well with pit floor inflow.

Predict steady mine inflow for a specified mine pit water level. The MineSteadyRadUnconfQ2 class uses the default Aq2dUnconf aquifer object with initial saturated thickness 100 for the upper unconfined aquifer.

aq

Upper aquifer object.

Type

obj

aq2kx

Horizontal hydraulic conductivity of lower aquifer (units L/T, default 1)

Type

float

aq2kz

Vertical hydraulic conductivity of lower aquifer (units L/T, default 1)

Type

float

class Aq2dUnconf(*K=1, Sy=0.1, B=10, bot=0, name='Aq2dUnconf class'*)

Bases: *Aquifer*

2D unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D unconfined aquifer with infinite lateral extent and unconfined storage.

The default Aq2dUnconf object has hydraulic conductivity K=1, specific yield Sy=0.1, static saturated thickness B=10 and aquifer bottom (datum) elevation bot=0. Exceptions will occur if invalid values are provided for K, Sy or B.

The .info and .draw methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = True

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

property D

Depth of mine pit lake (units L, default 0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property R

Groundwater recharge rate (units L/T, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

dd(*n=25, plot=True, csv="", xlsx=""*)

Evaluate distance-drawdown (units L).

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting `plot=False`.

Parameters

- **n** (*int*) – Number of radial values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default ‘’).
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default ‘’).

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level (units L, default 10).

Type

float

dr(*r*)

Drawdown at radius *r* (units L).

Parameters

r (*float*) – radius at which to evaluate drawdown (units L).

draw(*dw=8*)

Display the definition diagram.

Parameters

dw (*float*) – Width of figure (default 8.0).

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

hr(*r*)

Head at radius *r* (units L).

Parameters
r (*float*) – radius at which to evaluate head (units L).

info()

Print the solution information.

property qp

Total mine pit inflow rate (units L³/T).

Type
float

property qp1

Mine pit inflow rate from upper aquifer (units L³/T).

Type
float

property qp2

Mine pit inflow rate from lower aquifer (units L³/T).

Type
float

property ri

Radius of influence (units L).

Type
float

property rp

Equivalent radius of mine pit (units L, default 100).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

class pygaf.solutions.mine_flow.MineSteadyStripLeakyDD

Bases: object

Steady, leaky 1D flow to mine pit wall.

Predict steady mine drawdown for a specified mine pit inflow. The MineSteadyStripLeakyDD class uses the default Aq2dLeaky aquifer object with aquifer thickness 100 for the lower confined aquifer.

aq

Aquifer object.

Type
obj

class Aq2dLeaky($K=1$, $Ss=0.0001$, $B=10$, $bot=0$, $Kleak=1e-05$, $Bleak=10$, $name='Aq2dLeaky\ class'$)

Bases: [Aquifer](#)

2D leaky aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D leaky aquifer with infinite lateral extent and confined storage.

The default Aq2dLeak object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer bottom (datum) elevation $bot=0$, leaky hydraulic conductivity $Kleak=0.00001$ and leaky thickness $Bleak=10$. Exceptions will occur if invalid values are provided for K , Ss , B , $Kleak$ or $Bleak$.

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Bleak

Leaky layer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L^2/T , default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Kleak

Leaky layer hydraulic conductivity (units L/T , default 0.00001).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = True

is_semifinite = False

is_unconfined = False

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property beta

Solution term (units L).

Type

float

dd(n=25, plot=True, csv="", xlsx="")

Evaluate distance-drawdown (units L).

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting plot=False.

Parameters

- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).

- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default “”).
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default “”).

Returns

Results in a pandas dataframe.

property dp

Mine pit drawdown (units L).

Type

float

draw(dw=8)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dx(x)

Drawdown at specified distance from pit wall (unit L).

Parameters

x (*float*) – Distance from pit wall (units L).

property h0

Initial groundwater head (units L, default 120).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property hp

Mine pit head (units L).

Type

float

hx(x)

Head at specified distance from pit wall (units L).

Parameters

x (*float*) – Distance from pit wall (units L).

info()

Print the solution information.

property qp

Mine pit inflow rate (units L/T, default 0.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property xi

Length of influence defined where drawdown is equal to 0.1% of initial aquifer head (units L).

Type

float

class pygaf.solutions.mine_flow.MineSteadyStripLeakyQ

Bases: object

Steady, leaky 1D flow to mine pit wall.

Predict steady mine inflow for a specified mine pit water level. The MineSteadyStripLeakyQ class uses the default Aq2dLeaky aquifer object with aquifer thickness 100 for the lower confined aquifer.

aq

Aquifer object.

Type

obj

class Aq2dLeaky(*K=1, Ss=0.0001, B=10, bot=0, Kleak=1e-05, Bleak=10, name='Aq2dLeaky class'*)

Bases: [Aquifer](#)

2D leaky aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D leaky aquifer with infinite lateral extent and confined storage.

The default Aq2dLeak object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer bottom (datum) elevation $bot=0$, leaky hydraulic conductivity $Kleak=0.00001$ and leaky thickness $Bleak=10$. Exceptions will occur if invalid values are provided for K , Ss , B , $Kleak$ or $Bleak$.

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Bleak

Leaky layer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L^2/T , default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Kleak

Leaky layer hydraulic conductivity (units L/T, default 0.00001).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = True

is_semifinite = False

is_unconfined = False

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property beta

Solution term (units L).

Type

float

dd(*n=25, plot=True, csv="", xlsx=""*)

Evaluate distance-drawdown (units L).

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting `plot=False`.

Parameters

- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level from initial water table (units L, default 10).

Type

float

draw(*dw=8*)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dx(*x*)

Drawdown at specified distance from pit wall (unit L).

Parameters

x (*float*) – Distance from pit wall (units L).

property h0

Initial groundwater head (units L, default 120).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hx(*x*)

Head at specified distance from pit wall (units L).

Parameters

x (*float*) – Distance from pit wall (units L).

info()

Print the solution information.

property qp

Mine pit inflow rate (units L³/T).

Type

float

property xi

Length of influence defined where drawdown is equal to 0.1% of initial aquifer head (units L).

Type

float

class pygaf.solutions.mine_flow.**MineSteadyStripUnconfQ**

Bases: object

Steady, unconfined 1D flow to mine pit wall.

Predict steady mine inflow for a specified mine pit water level. The MineSteadyStripUnconfQ class uses the default Aq2dUnconf aquifer object with initial saturated thickness 100.

aq

Aquifer object.

Type

obj

class Aq2dConf(*K=1, Ss=0.0001, B=10, bot=0, name='Aq2dConf class'*)

Bases: [Aquifer](#)

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters**dw** (*float*) – Width of figure (default 6.0).**info()**

Print the aquifer information.

is_1d = False**is_2d = True****is_confined = True****is_finite = False****is_heterogeneous = False****is_homogeneous = True****is_infinite = True****is_leaky = False****is_semifinite = False****is_unconfined = False****property R**

Groundwater recharge rate (units L/T, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

dd(*n=25, plot=True, csv="", xlsx=""*)

Evaluate distance-drawdown (units L).

Evaluate steady state drawdown at specified distances from the mine pit wall. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting `plot=False`.

Parameters

- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level from initial water table (units L, default 10).

Type

float

draw(*dw=8*)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dx(*x*)

Drawdown at specified distance from pit wall (unit L).

Parameters

x (*float*) – Distance from pit wall (units L).

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hx(*x*)

Head at specified distance from pit wall (units L).

Parameters

x (*float*) – Distance from pit wall (units L).

info()

Print the solution information.

property qp

Mine pit inflow rate (units L³/T).

Type

float

property xi

Length of influence (units L).

Type

float

class pygaf.solutions.mine_flow.MineTransRadConfDD

Bases: object

Transient, radial, confined flow to a large diameter well.

Predict transient drawdown for a specified mine pit inflow. The MineTransRadConfDD class uses the default Aq2dConf aquifer object with aquifer thickness 100 for the lower confined aquifer.

aq

Lower aquifer object.

Type

obj

class Aq2dConf($K=1$, $Ss=0.0001$, $B=10$, $bot=0$, $name='Aq2dConf class'$)

Bases: [Aquifer](#)

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L^2/T , default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T , default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters**dw** (*float*) – Width of figure (default 6.0).**info()**

Print the aquifer information.

is_1d = False**is_2d** = True**is_confined** = True**is_finite** = False**is_heterogeneous** = False**is_homogeneous** = True**is_infinite** = True**is_leaky** = False**is_semifinite** = False**is_unconfined** = False**dd(t, n=25, plot=True, csv="", xlsx="")**

Evaluate radial-drawdown at specified time.

Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting plot=False.

Parameters

- **t** (*float*) – Time for evaluating drawdown (units T).
- **n** (*int*) – Number of radial values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

dp(*n*=25, *plot*=True, *csv*="", *xlsx*="")

Transient drawdown of mine pit water level.

Evaluate transient drawdown of the water level in the mine pit. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting *plot*=False.

Parameters

- **n** (*int*) – Number of radial values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp_targ

Mine pit drawdown target (units L, default 10).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property dp_targ_time

Time at which the mine pit drawdown target is reached (units T).

Type

float

draw(*dw*=8)

Display the drawing definition.

Parameters

- **dw** (*float*) – Width of figure (default 8.0).

drt(*r*, *t*)

Drawdown at specified radius and time (units L).

Parameters

- **r** (*float*) – radius (units L).
- **t** (*float*) – time (units T).

property h0

Initial groundwater head (units L, default 110).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hrt(*r*, *t*)

Head at specified radius and time (units L).

Parameters

- **r** (*float*) – radius (units L).

- t (*float*) – time (units T).

info()

Print the solution information.

property qp

Mine pit inflow rate (units L/T, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

ri(*t*)

Radius of influence at specified time.

Defined as radius within which change in storage is equal to 95% of total discharge.

Parameters

t (*float*) – time (units T).

property rp

Equivalent radius of mine pit (units L, default 100).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

class pygaf.solutions.mine_flow.MineTransStripConfQ

Bases: object

Transient, confined 1D flow to mine pit wall.

Predict transient inflow for a specified mine pit water level. The MineTransStripConfQ class uses the default Aq2dConf aquifer object with aquifer thickness 100.

aq

Aquifer object.

Type

obj

class Aq2dConf(*K=1, Ss=0.0001, B=10, bot=0, name='Aq2dConf class'*)

Bases: [Aquifer](#)

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = True

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = False

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

dd(*t*, *n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate distance-drawdown at specified time(units L).

Evaluate drawdown at specified distances from the mine pit wall at specified time. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting *plot*=False.

Parameters

- **t** (*float*) – Time (units T)
- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level (units L).

Type

float

draw(*dw*=8)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dxt(*x*, *t*)

Drawdown at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

property h0

Initial groundwater head (units L, default 110).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
float

hxt(*x*, *t*)

Head at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

info()

Print the solution information.

qp(*t*)

Inflow to mine pit at specified time (units L³/T).

Parameters

t (*float*) – time (units T)

qp_cum(*t*)

Cumulative inflow to mine pit at specified time (units L³).

Parameters

t (*float*) – time (units T)

xi(*t*)

Length of influence at specified time, defined where drawdown is equal to 0.1% of initial aquifer head (units L).

Parameters

t (*float*) – time (units T)

class pygaf.solutions.mine_flow.**MineTransStripLeakyQ**

Bases: object

Transient, leaky 1D flow to mine pit wall.

Predict transient inflow for a specified mine pit water level. The MineTransStripLeakyQ class uses the default Aq2dLeaky aquifer object with aquifer thickness 100.

aq

Aquifer object.

Type
obj

class Aq2dLeaky(*K=1*, *Ss=0.0001*, *B=10*, *bot=0*, *Kleak=1e-05*, *Bleak=10*, *name='Aq2dLeaky class'*)

Bases: [Aquifer](#)

2D leaky aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D leaky aquifer with infinite lateral extent and confined storage.

The default Aq2dLeak object has hydraulic conductivity *K*=1, specific storativity *Ss*=0.0001, aquifer saturated thickness *B*=10, aquifer bottom (datum) elevation *bot*=0, leaky hydraulic conductivity *Kleak*=0.00001 and leaky thickness *Bleak*=10. Exceptions will occur if invalid values are provided for *K*, *Ss*, *B*, *Kleak* or *Bleak*.

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Bleak

Leaky layer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property Kleak

Leaky layer hydraulic conductivity (units L/T, default 0.00001).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

```

is_1d = False
is_2d = True
is_confined = False
is_finite = False
is_heterogeneous = False
is_homogeneous = True
is_infinite = True
is_leaky = True
is_semifinite = False
is_unconfined = False

```

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property beta

Leakage factor.

Type

float

dd(*t*, *n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate distance-drawdown at specified time(units L).

Evaluate drawdown at specified distances from the mine pit wall at specified time. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting plot=False.

Parameters

- **t** (*float*) – Time (units T)
- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level (units L).

Type

float

draw(*dw*=8)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dxt(*x*, *t*)

Drawdown at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

property h0

Initial groundwater head (units L, default 110).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hxt(*x*, *t*)

Head at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

info()

Print the solution information.

qp(*t*)

Inflow to mine pit at specified time (units L³/T).

Parameters

t (*float*) – time (units T)

property qp_steady

Inflow to mine pit at steady state (units L³/T).

Type

float

property xi_steady

Length of influence at specified time, defined where drawdown is equal to 0.1% of initial aquifer head (units L).

class pygaf.solutions.mine_flow.MineTransStripUnconfQ

Bases: object

Transient, unconfined 1D flow to mine pit wall.

Predict transient inflow for a specified mine pit water level. The MineTransStripUnconfQ class uses the default Aq2dUnconf aquifer object initial saturated thickness 100.

aq

Aquifer object.

Type

obj

class Aq2dUnconf(*K=1, Sy=0.1, B=10, bot=0, name='Aq2dUnconf class'*)

Bases: [Aquifer](#)

2D unconfined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 2D unconfined aquifer with infinite lateral extent and unconfined storage.

The default Aq2dUnconf object has hydraulic conductivity $K=1$, specific yield $Sy=0.1$, static saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Sy or B .

The .info and .draw methods display the aquifer information and diagram.

Sy

Aquifer specific yield (units 1, default 0.1).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 0.1).

Type

float

property Sy

Aquifer specific yield (units 1, default 0.1).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*dw*=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = False

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = True

property swl

Aquifer static water table elevation (units L reduced level, default 10.0).

Type

float

property Y

Length of mine strip (units L, default 1000).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

dd(*t*, *n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate distance-drawdown at specified time(units L).

Evaluate drawdown at specified distances from the mine pit wall at specified time. Results are returned in a Pandas dataframe. A drawdown graph is displayed as default and can be suppressed by setting *plot*=False.

Parameters

- **t** (*float*) – Time (units T)
- **n** (*int*) – Number of values for evaluating drawdown (default 25).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

property dp

Drawdown of mine pit water level (units L).

Type

float

draw(*dw=8*)

Display the drawing definition.

Parameters

dw (*float*) – Width of figure (default 8.0).

dxt(*x, t*)

Drawdown at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

property hp

Mine pit water level (units L, default 90).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

hxt(*x, t*)

Head at specified distance and time (units L).

Parameters

- **x** (*float*) – distance from pit wall (units L).
- **t** (*float*) – time (units T).

info()

Print the solution information.

qp(*t*)

Inflow to mine pit at specified time (units L³/T).

Parameters

t (*float*) – time (units T)

qp_cum(*t*)

Cumulative inflow to mine pit at specified time (units L³).

Parameters

t (*float*) – time (units T)

xi(*t*)

Length of influence at specified time, defined where drawdown is equal to 0.1% of initial aquifer head (units L).

Parameters

t (*float*) – time (units T)

1.3.3 Steady state 1D aquifer flow

class pygaf.solutions.steady_flow.Steady1dConfFlow

Bases: object

Steady state flow in a 1D confined aquifer.

The Steady1dConfFlow class uses the default Aq1dFiniteConf aquifer object, default SteadyBC type 2 boundary condition object at $x=0$ and default SteadyBC type 1 boundary condition object at $x=L$.

aq

Aquifer object.

Type

obj

bc0

SteadyBC object at $x=0$ (default type=2).

Type

obj

bcL

SteadyBC object at $x=L$ (default type=1).

Type

obj

R

Groundwater recharge rate (units L/T, default 0.0)

Type

float

class Aq1dFiniteConf($K=1$, $Ss=0.0001$, $B=10$, $L=1000$, $bot=0$, $name='Aq1dFiniteConf class'$)

Bases: [Aquifer](#)

1D, finite confined aquifer subclass.

A subclass of the Aquifer class defining a horizontal, 1D confined aquifer with finite lateral extent and confined storage.

The default Aq1dFiniteConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$, aquifer length $L=1000$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss , B or L .

The .info and .draw methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default $1.0e-4$).

Type

float

L

Aquifer length (units L, default 1000.0).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property L

Aquifer length (units L, default 1000.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = True

is_2d = False

is_confined = True

is_finite = True

is_heterogeneous = False

is_homogeneous = True

is_infinite = False

is_leaky = False

is_radial = False

is_semifinite = False

is_unconfined = False

class SteadyBC(*type=2, head=10.0, flow=0.0, cond=0.0*)

Bases: object

Boundary condition for steady state groundwater flow.

The boundary condition can be type 1, 2 or 3 (see the ‘type’ attribute). An exception occurs if any other value is specified for type. The default SteadyBC object is a type 2 boundary condition with flow=0.0, which is equivalent to a no-flow boundary. The default type 1 Boundary has head=10.0 (same as the default head of all Aquifer classes). The default type 3 boundary condition has head=10.0 and cond=0.0, which is equivalent to a no-flow condition.

Note a utilities function is available to calculate conductance values

type

Boundary condition type; choices are type=1 (Dirichlet, first-type or constant head), type=2 (Neumann, second-type or constant flow) or type=3 (Cauchy or general head).

Type
int

head

Value of head at the boundary for type 1 and value of

Type
float

external head for type 2

Type
units L, default 10.0

flow

value of normal flow at boundary for type 2 (units L/T, default 0.0).

Type
float

cond

Value of conductance for type 3 (units L²/T, default 0.0).

Type
float

info()

Print the solution information.

property type

Boundary condition type.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type
int

property value

Boundary condition value(s).

Type

dic

bc_t1_t1(*H0*, *HL*, *L*, *T*, *R*, *x*)Aquifer solution for type 1 bc at $x=0$ and type 1 bc at $x=L$.**bc_t1_t2**(*H*, *Q*, *L*, *T*, *R*, *x*)Aquifer solution for type 1 bc at $x=0$ and type 2 bc at $x=L$.**bc_t2_t1**(*H*, *Q*, *L*, *T*, *R*, *x*)Aquifer solution for type 2 bc at $x=0$ and type 1 bc at $x=L$.**h**(*n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate aquifer head.

Parameters

- **n** (*int*) – Number of evenly-spaced x values at which to evaluate aquifer head.
- **plot** (*bool*) – Display a plot of the results (default True).
- **csv** (*str*) – Filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Pandas dataframe containing head values.

h_grad(*n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate aquifer head gradient.

Parameters

- **n** (*int*) – Number of evenly-spaced x values at which to evaluate aquifer flow.
- **plot** (*bool*) – Display a plot of the results (default True).
- **csv** (*str*) – Filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Pandas dataframe containing head values.

info()

Print the solution information.

q(*n*=25, *plot*=True, *csv*="", *xlsx*="")

Evaluate aquifer flow.

Parameters

- **n** (*int*) – Number of evenly-spaced x values at which to evaluate aquifer flow.
- **plot** (*bool*) – Display a plot of the results (default True).
- **csv** (*str*) – Filepath for export of results to csv file; results are exported if the string is not empty (default "").

- **xlsx** (*str*) – Filepath for export of result to xlsx file; results are exported if the string is not empty (default ‘’).

Returns

Pandas dataframe containing head values.

property types

Boundary condition types.

Type

int

1.3.4 Theis (1935) radial flow

class `pygaf.solutions.theis_1935.TheisWell`

Bases: `object`

Theis (1935) radial flow solution.

The default TheisWell object uses the default Aq2dConf and WellGrid classes. The WellGrid class includes the default Well class. Methods include radius of influence `.ri`, transient drawdown at a point `.dd` and grid-contoured drawdown at specified time `.dd_grid`.

aq

Aq2dConf aquifer object.

Type

obj

grd

WellGrid object.

Type

obj

qf

Fraction of pumped volume used for calculating radius of influence (default 0.99).

Type

float

class `Aq2dConf(K=1, Ss=0.0001, B=10, bot=0, name='Aq2dConf class')`

Bases: [*Aquifer*](#)

2D confined aquifer class.

A subclass of the Aquifer class defining a horizontal, 2D confined aquifer with infinite lateral extent and confined storage.

The default Aq2dConf object has hydraulic conductivity $K=1$, specific storativity $Ss=0.0001$, aquifer saturated thickness $B=10$ and aquifer bottom (datum) elevation $bot=0$. Exceptions will occur if invalid values are provided for K , Ss or B .

The `.info` and `.draw` methods display the aquifer information and diagram.

Ss

Aquifer specific storativity (units 1/L, default 1.0e-4).

Type

float

property B

Aquifer thickness (units L, default 10.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property D

Aquifer diffusivity (units L²/T, default 1.0e+4).

Type

float

property K

Aquifer hydraulic conductivity (units L/T, default 1.0).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property S

Aquifer storage coefficient (units 1, default 1.0e-3).

Type

float

property Ss

aquifer specific storativity (units 1/L, default 1.0e-4).

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(dw=6)

Display a drawing of the aquifer.

Parameters

dw (*float*) – Width of figure (default 6.0).

info()

Print the aquifer information.

is_1d = False

is_2d = True

is_confined = True

is_finite = False

is_heterogeneous = False

is_homogeneous = True

is_infinite = True

is_leaky = False

is_semifinite = False

is_unconfined = False

class WellGrid(*gr=100, gd=21*)

Bases: object

Square grid class with regular spacing and well at grid center.

The default WellGrid object has radius *gr*=100 defining the square's extent and grid density *gd*=21. An exception occurs if the grid radius is not positive. Grid density defines the numbers of rows and columns comprising the grid; thus, the default grid has 21 rows x 21 cols = 441 grid points. Minimum grid density is limited to 11 (121 grid points) and maximum grid density is limited to 41 (1681 grid points). Values for *gd* outside of these constraints are re-set to the minimum or maximum values as appropriate.

The *.pts* property returns the grid points attributes including local x-y coordinates, world x-y coordinates and radius values relative to the well location. The *.info* method displays grid information and the *.draw* method displays a plot of the grid in either local or world coordinates.

gr

Radius defining the extent of the solution grid (units L, default 100.0).

Type

float

gd

Grid density defining the number of rows and columns; minimum and maximum constraints are enforced (default 21).

Type

int

class Well(*x=0.0, y=0.0, r=0.05, pf=1.0, name='Steady state non-flowing well'*)

Bases: object

Non-flowing well.

x

Well x coordinate (units L, default 0.0).

Type

float

y

Well y coordinate (units L, default 0.0).

Type

float

r

Well radius (units L, default 0.05).

Type

float

pf

Well penetration depth (fraction of aquifer depth, default 1.0).

Type

float

name

Well name (default '').

Type

str

info()

Print the well information.

property pf

Well penetration depth.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property r

Well radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

draw(*local=False*)

Draw the grid points.

Parameters

local (*bool*) – Display the grid plot in local coordinates with the well at 0, 0 (default False).

property gr

Grid radius.

Setter method checks for valid values and triggers an exception if invalid values are specified.

Type

float

property grdim

Number of grid rows and columns.

Type

int

info()

Print the well grid information.

property npts

Number of grid points.

Type

int

property pts

grid point attributes including local grid point coordinates, world grid point coordinates and radius values of grid points relative to the well center.

Type

pandas dataframe

dd(*t=[1], r=[1], q=-1000.0, plot=True, csv="", xlsx=""*)

Evaluate drawdown at specified radii and times.

Evaluate drawdown at each radius and time specified in the lists *t* and *r* and for well rate *q*. Defaults are *t*=[1.0], *r*=[1.0] and *q*=-1000.0. A drawdown graph is displayed as default and can be suppressed by setting *plot=False*.

A pandas dataframe of drawdown values is returned with time as the row index and radius values as the column headers. Results can be exported to csv and Excel files by setting non-blank filename strings for the *.csv* and *.xlsx* attributes. Filenames can be supplied with or without file extensions, which are added if omitted.

Parameters

- **t** (*float*) – List of times to evaluate drawdown (default [1.0]).
- **r** (*float*) – List of radii to evaluate drawdown (default [1.0]).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

dd_grid(*t=1.0, q=-1000.0, plot=True, local=False, csv="", xlsx=""*)

Evaluate drawdown on a regular grid.

Evaluate drawdown on a grid of points at specified time and well rate. Default values are *t=1.0* and *q=-1000*. Unless otherwise specified, a default WellGrid object is used; it can be accessed and adjusted via the `.grid.gr` (grid radius) and `.grid.gd` (grid density) attributes.

Results are returned in a Pandas dataframe with column x-coord, y-coord and drawdown value. A drawdown graph is displayed as default and can be suppressed by setting *plot=False*.

Parameters

- **t** (*float*) – Time of drawdown (default 1.0).
- **plot** (*bool*) – Display a plot of results (default True).
- **local** (*bool*) – Display the results in 'local' coordinates with the well at coordinates 0.0, 0.0 (Default False).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default "").
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default "").

Returns

Results in a pandas dataframe.

disp(*r, S, T, t, Q*)

Drawdown displacement.

info()

Print the solution information.

property qfp

Fraction of pumped volume.

Type

float

ri(*t=[1.0], q=-1000, plot=True, csv="", xlsx=""*)

Calculate radius of influence at specified times.

Radius of influence is defined as the radius from within which a specified fraction *qf* of the pumped volume has been drawn. The default value for *qf* is 0.99, corresponding to the radius from which 99% of the pumped volume has been drawn.

Time for calculating *ri* are provided in a list. A results graph is displayed as default and can be suppressed by setting *plot=False*. A pandas dataframe is returned with time as the row index and *ri* as a column.

Results can be exported to csv and Excel files by setting non-blank filename strings for the .csv and .xlsx attributes. Filenames can be supplied with or without file extensions, which are added if omitted.

Parameters

- **t** (*float*) – List of times to evaluate radius of influence (default [1.0]).
- **plot** (*bool*) – Display a plot of results (default True).
- **csv** (*str*) – Full filepath for export of results to csv file; results are exported if the string is not empty (default “”).
- **xlsx** (*str*) – Full filepath for export of result to xlsx file; results are exported if the string is not empty (default “”).

Returns

Results in pandas dataframe.

rinf(T, S, t, qf)

Radius of influence.

PYTHON MODULE INDEX

p

- `pygaf.aquifers`, 3
- `pygaf.basins`, 15
- `pygaf.bcs`, 18
- `pygaf.grids`, 19
- `pygaf.solutions.glover_1960`, 30
- `pygaf.solutions.mine_flow`, 36
- `pygaf.solutions.steady_flow`, 68
- `pygaf.solutions.theis_1935`, 72
- `pygaf.stresses`, 23
- `pygaf.utils`, 24
- `pygaf.wells`, 26

INDEX

A

[add_constant_to_list\(\)](#) (in module `pygaf.utils`), 24
[aq](#) (`pygaf.solutions.glover_1960.GloverRectBasinSteady` attribute), 30
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyRadLeakyDD` attribute), 36
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ` attribute), 39
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2` attribute), 42
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD` attribute), 45
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ` attribute), 49
[aq](#) (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ` attribute), 52
[aq](#) (`pygaf.solutions.mine_flow.MineTransRadConfDD` attribute), 55
[aq](#) (`pygaf.solutions.mine_flow.MineTransStripConfQ` attribute), 58
[aq](#) (`pygaf.solutions.mine_flow.MineTransStripLeakyQ` attribute), 61
[aq](#) (`pygaf.solutions.mine_flow.MineTransStripUnconfQ` attribute), 65
[aq](#) (`pygaf.solutions.steady_flow.Steady1dConfFlow` attribute), 68
[aq](#) (`pygaf.solutions.theis_1935.TheisWell` attribute), 72
[Aq1dFiniteConf](#) (class in `pygaf.aquifers`), 3
[Aq1dFiniteUnconf](#) (class in `pygaf.aquifers`), 5
[Aq1dSemi finiteConf](#) (class in `pygaf.aquifers`), 6
[Aq1dSemi finiteUnconf](#) (class in `pygaf.aquifers`), 8
[Aq2dConf](#) (class in `pygaf.aquifers`), 9
[Aq2dLeaky](#) (class in `pygaf.aquifers`), 11
[Aq2dUnconf](#) (class in `pygaf.aquifers`), 13
[aq2kx](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2B` attribute), 42
[aq2kz](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2B` attribute), 42
[Aquifer](#) (class in `pygaf.aquifers`), 14
[area](#) (`pygaf.basins.CircBasin` property), 15
[area](#) (`pygaf.basins.RectBasin` property), 16
[area](#) (`pygaf.grids.BasinGrid.RectBasin` property), 20

[area](#) (`pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin` property), 33

B

[B](#) (`pygaf.aquifers.Aq1dFiniteConf` property), 3
[B](#) (`pygaf.aquifers.Aq1dFiniteUnconf` property), 5
[B](#) (`pygaf.aquifers.Aq1dSemi finiteConf` property), 7
[B](#) (`pygaf.aquifers.Aq1dSemi finiteUnconf` property), 8
[B](#) (`pygaf.aquifers.Aq2dConf` property), 10
[B](#) (`pygaf.aquifers.Aq2dLeaky` property), 11
[B](#) (`pygaf.aquifers.Aq2dUnconf` property), 13
[B](#) (`pygaf.aquifers.Aquifer` attribute), 14
[B](#) (`pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf` property), 31
[B](#) (`pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky` property), 36
[B](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf` property), 39
[B](#) (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf` property), 43
[B](#) (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky` property), 46
[B](#) (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky` property), 49
[B](#) (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf` property), 52
[B](#) (`pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf` property), 55
[B](#) (`pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf` property), 58
[B](#) (`pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky` property), 62
[B](#) (`pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf` property), 65
[B](#) (`pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf` property), 68
[basin](#) (`pygaf.solutions.glover_1960.GloverRectBasinSteady` attribute), 31
[BasinGrid](#) (class in `pygaf.grids`), 19
[BasinGrid.RectBasin](#) (class in `pygaf.grids`), 19

bc0 (pygaf.solutions.steady_flow.Steady1dConfFlow attribute), 68
 bc_t1_t1() (pygaf.solutions.steady_flow.Steady1dConfFlow method), 71
 bc_t1_t2() (pygaf.solutions.steady_flow.Steady1dConfFlow method), 71
 bc_t2_t1() (pygaf.solutions.steady_flow.Steady1dConfFlow method), 71
 bcL (pygaf.solutions.steady_flow.Steady1dConfFlow attribute), 68
 beta (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD property), 47
 beta (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ property), 50
 beta (pygaf.solutions.mine_flow.MineTransStripLeakyQ property), 63
 Bleak (pygaf.aquifers.Aq2dLeaky property), 11
 Bleak (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 36
 Bleak (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
 Bleak (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 49
 Bleak (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
 bot (pygaf.aquifers.Aquifer attribute), 14
C
 CircBasin (class in pygaf.basins), 15
 cond (pygaf.bcs.SteadyBC attribute), 18
 cond (pygaf.solutions.steady_flow.Steady1dConfFlow.SteadyBC attribute), 70
 conductance() (in module pygaf.utils), 24
 cx (pygaf.basins.CircBasin attribute), 15
 cx (pygaf.basins.RectBasin attribute), 16
 cx (pygaf.grids.BasinGrid.RectBasin attribute), 19
 cx (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute), 33
 cy (pygaf.basins.CircBasin attribute), 15
 cy (pygaf.basins.RectBasin attribute), 16
 cy (pygaf.grids.BasinGrid.RectBasin attribute), 19
 cy (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute), 33
D
 D (pygaf.aquifers.Aq1dFiniteConf property), 4
 D (pygaf.aquifers.Aq1dSemifiniteConf property), 7
 D (pygaf.aquifers.Aq2dConf property), 10
 D (pygaf.aquifers.Aq2dLeaky property), 11
 D (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf property), 31
 D (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 37
 D (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 property), 44
 D (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
 D (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 49
 D (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dConf property), 52
 D (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf property), 55
 D (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf property), 59
 D (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
 D (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf property), 69
 D (pygaf.solutions.theis_1935.TheisWell.Aq2dConf property), 73
 dd() (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD method), 38
 dd() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ method), 41
 dd() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 method), 44
 dd() (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD method), 47
 dd() (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ method), 50
 dd() (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ method), 53
 dd() (pygaf.solutions.mine_flow.MineTransRadConfDD method), 56
 dd() (pygaf.solutions.mine_flow.MineTransStripConfQ method), 60
 dd() (pygaf.solutions.mine_flow.MineTransStripLeakyQ method), 63
 dd() (pygaf.solutions.mine_flow.MineTransStripUnconfQ method), 66
 dd() (pygaf.solutions.theis_1935.TheisWell method), 75
 dd_grid() (pygaf.solutions.theis_1935.TheisWell method), 76
 dec2rad() (in module pygaf.utils), 25
 diam (pygaf.basins.CircBasin attribute), 15
 diam (pygaf.basins.CircBasin property), 15
 disp() (pygaf.solutions.theis_1935.TheisWell method), 76
 display_image() (in module pygaf.utils), 25
 dp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ property), 41
 dp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 property), 44
 dp (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD property), 48
 dp (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ property), 62

- property), 51
- dp (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ property), 54
- dp (pygaf.solutions.mine_flow.MineTransStripConfQ property), 60
- dp (pygaf.solutions.mine_flow.MineTransStripLeakyQ property), 63
- dp (pygaf.solutions.mine_flow.MineTransStripUnconfQ property), 67
- dp() (pygaf.solutions.mine_flow.MineTransRadConfDD method), 56
- dp_targ (pygaf.solutions.mine_flow.MineTransRadConfDD property), 57
- dp_targ_time (pygaf.solutions.mine_flow.MineTransRadConfDD property), 57
- dr() (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD method), 38
- dr() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ method), 41
- dr() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 method), 44
- draw() (pygaf.aquifers.Aq1dFiniteConf method), 4
- draw() (pygaf.aquifers.Aq1dFiniteUnconf method), 6
- draw() (pygaf.aquifers.Aq1dSemifiniteConf method), 7
- draw() (pygaf.aquifers.Aq1dSemifiniteUnconf method), 9
- draw() (pygaf.aquifers.Aq2dConf method), 10
- draw() (pygaf.aquifers.Aq2dLeaky method), 12
- draw() (pygaf.aquifers.Aq2dUnconf method), 13
- draw() (pygaf.basins.CircBasin method), 16
- draw() (pygaf.basins.RectBasin method), 17
- draw() (pygaf.grids.BasinGrid method), 21
- draw() (pygaf.grids.BasinGrid.RectBasin method), 20
- draw() (pygaf.grids.WellGrid method), 22
- draw() (pygaf.solutions.glover_1960.GloverRectBasinSteadyAq2dConf method), 32
- draw() (pygaf.solutions.glover_1960.GloverRectBasinSteadyAq2dLeaky method), 34
- draw() (pygaf.solutions.glover_1960.GloverRectBasinSteadyAq2dUnconf method), 33
- draw() (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD method), 38
- draw() (pygaf.solutions.mine_flow.MineSteadyRadLeakyDDAq2dLeaky method), 37
- draw() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ method), 41
- draw() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQAq2dUnconf method), 40
- draw() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 method), 44
- draw() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf method), 43
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD method), 48
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyDDAq2dLeaky method), 47
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ method), 51
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyQAq2dLeaky method), 50
- draw() (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ method), 54
- draw() (pygaf.solutions.mine_flow.MineSteadyStripUnconfQAq2dConf method), 53
- draw() (pygaf.solutions.mine_flow.MineTransRadConfDD method), 57
- draw() (pygaf.solutions.mine_flow.MineTransRadConfDDAq2dConf method), 56
- draw() (pygaf.solutions.mine_flow.MineTransStripConfQ method), 60
- draw() (pygaf.solutions.mine_flow.MineTransStripConfQAq2dConf method), 59
- draw() (pygaf.solutions.mine_flow.MineTransStripLeakyQ method), 63
- draw() (pygaf.solutions.mine_flow.MineTransStripLeakyQAq2dLeaky method), 62
- draw() (pygaf.solutions.mine_flow.MineTransStripUnconfQ method), 67
- draw() (pygaf.solutions.mine_flow.MineTransStripUnconfQAq2dUnconf method), 65
- draw() (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf method), 69
- draw() (pygaf.solutions.theis_1935.TheisWell.Aq2dConf method), 73
- draw() (pygaf.solutions.theis_1935.TheisWell.WellGrid method), 75
- drt() (pygaf.solutions.mine_flow.MineTransRadConfDD method), 57
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD method), 48
- draw() (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ method), 51
- draw() (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ method), 54
- draw() (pygaf.solutions.mine_flow.MineTransStripConfQ method), 60
- draw() (pygaf.solutions.mine_flow.MineTransStripLeakyQ method), 64
- draw() (pygaf.solutions.mine_flow.MineTransStripUnconfQ method), 67
- ## F
- flow (pygaf.bcs.SteadyBC attribute), 18
- flow (pygaf.solutions.steady_flow.Steady1dConfFlow.SteadyBC attribute), 70
- from_csv (pygaf.stresses.StressSeries attribute), 24
- from_csv (pygaf.wells.TransientWell.StressSeries attribute), 28

G

- gd (pygaf.grids.BasinGrid attribute), 19
- gd (pygaf.grids.WellGrid attribute), 21
- gd (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid attribute), 32
- gd (pygaf.solutions.theis_1935.TheisWell.WellGrid attribute), 74
- GloverRectBasinSteady (class in pygaf.solutions.glover_1960), 30
- GloverRectBasinSteady.Aq2dConf (class in pygaf.solutions.glover_1960), 31
- GloverRectBasinSteady.BasinGrid (class in pygaf.solutions.glover_1960), 32
- GloverRectBasinSteady.BasinGrid.RectBasin (class in pygaf.solutions.glover_1960), 32
- gr (pygaf.grids.BasinGrid attribute), 19
- gr (pygaf.grids.BasinGrid property), 21
- gr (pygaf.grids.WellGrid attribute), 21
- gr (pygaf.grids.WellGrid property), 23
- gr (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid attribute), 32
- gr (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid property), 34
- gr (pygaf.solutions.theis_1935.TheisWell.WellGrid attribute), 74
- gr (pygaf.solutions.theis_1935.TheisWell.WellGrid property), 75
- grd (pygaf.solutions.theis_1935.TheisWell attribute), 72
- grdim (pygaf.grids.BasinGrid property), 21
- grdim (pygaf.grids.WellGrid property), 23
- grdim (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid property), 34
- grdim (pygaf.solutions.theis_1935.TheisWell.WellGrid property), 75
- H
- h() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 35
- h() (pygaf.solutions.steady_flow.Steady1dConfFlow method), 71
- h0 (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD property), 38
- h0 (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD property), 48
- h0 (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ property), 51
- h0 (pygaf.solutions.mine_flow.MineTransRadConfDD property), 57
- h0 (pygaf.solutions.mine_flow.MineTransStripConfQ property), 60
- h0 (pygaf.solutions.mine_flow.MineTransStripLeakyQ property), 64
- h_grad() (pygaf.solutions.steady_flow.Steady1dConfFlow method), 71
- head (pygaf.bcs.SteadyBC attribute), 18
- head (pygaf.solutions.steady_flow.Steady1dConfFlow.SteadyBC attribute), 70
- hp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ property), 41
- hp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 property), 44
- hp (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD property), 48
- hp (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ property), 51
- hp (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ property), 54
- hp (pygaf.solutions.mine_flow.MineTransStripConfQ property), 60
- hp (pygaf.solutions.mine_flow.MineTransStripLeakyQ property), 64
- hp (pygaf.solutions.mine_flow.MineTransStripUnconfQ property), 67
- hr() (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD method), 38
- hr() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ method), 41
- hr() (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 method), 45
- hrt() (pygaf.solutions.mine_flow.MineTransRadConfDD method), 57
- hx() (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD method), 48
- hx() (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ method), 51
- hx() (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ method), 54
- hxt() (pygaf.solutions.mine_flow.MineTransStripConfQ method), 61
- hxt() (pygaf.solutions.mine_flow.MineTransStripLeakyQ method), 64
- hxt() (pygaf.solutions.mine_flow.MineTransStripUnconfQ method), 67
- I
- impress() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 35
- impress_grid() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 35
- info() (pygaf.aquifers.Aq1dFiniteConf method), 4
- info() (pygaf.aquifers.Aq1dFiniteUnconf method), 6
- info() (pygaf.aquifers.Aq1dSemifiniteConf method), 7
- info() (pygaf.aquifers.Aq1dSemifiniteUnconf method), 9
- info() (pygaf.aquifers.Aq2dConf method), 10
- info() (pygaf.aquifers.Aq2dLeaky method), 12
- info() (pygaf.aquifers.Aq2dUnconf method), 13
- info() (pygaf.basins.CircBasin method), 16
- info() (pygaf.basins.RectBasin method), 17

- `info()` (`pygaf.bcs.SteadyBC` method), 18
`info()` (`pygaf.grids.BasinGrid` method), 21
`info()` (`pygaf.grids.BasinGrid.RectBasin` method), 20
`info()` (`pygaf.grids.WellGrid` method), 23
`info()` (`pygaf.grids.WellGrid.Well` method), 22
`info()` (`pygaf.solutions.glover_1960.GloverRectBasinSteady` method), 35
`info()` (`pygaf.solutions.glover_1960.GloverRectBasinSteady` method), 32
`info()` (`pygaf.solutions.glover_1960.GloverRectBasinSteady` method), 34
`info()` (`pygaf.solutions.glover_1960.GloverRectBasinSteady` method), 33
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadLeakyDD` method), 38
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadLeakyDD` method), 37
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ` method), 41
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ` method), 40
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ` method), 45
`info()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ` method), 43
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD` method), 48
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD` method), 47
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ` method), 51
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ` method), 50
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ` method), 54
`info()` (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ` method), 53
`info()` (`pygaf.solutions.mine_flow.MineTransRadConfDD` method), 58
`info()` (`pygaf.solutions.mine_flow.MineTransRadConfDD` method), 56
`info()` (`pygaf.solutions.mine_flow.MineTransStripConfQ` method), 61
`info()` (`pygaf.solutions.mine_flow.MineTransStripConfQ` method), 59
`info()` (`pygaf.solutions.mine_flow.MineTransStripLeakyQ` method), 64
`info()` (`pygaf.solutions.mine_flow.MineTransStripLeakyQ` method), 62
`info()` (`pygaf.solutions.mine_flow.MineTransStripUnconfQ` method), 67
`info()` (`pygaf.solutions.mine_flow.MineTransStripUnconfQ` method), 66
`info()` (`pygaf.solutions.steady_flow.Steady1dConfFlow` method), 71
`info()` (`pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf` method), 69
`info()` (`pygaf.solutions.steady_flow.Steady1dConfFlow.SteadyBC` method), 70
`info()` (`pygaf.solutions.theis_1935.TheisWell` method), 76
`info()` (`pygaf.solutions.theis_1935.TheisWell.Aq2dConf` method), 73
`info()` (`pygaf.solutions.theis_1935.TheisWell.WellGrid` method), 75
`info()` (`pygaf.solutions.theis_1935.TheisWell.WellGrid.Well` method), 74
`info()` (`pygaf.wells.SteadyWell` method), 27
`info()` (`pygaf.wells.TransientWell` method), 29
`is_1d()` (`pygaf.wells.Well` method), 30
`is_1d()` (`pygaf.aquifers.Aq1dFiniteConf` attribute), 4
`is_1d()` (`pygaf.aquifers.Aq1dFiniteUnconf` attribute), 6
`is_1d()` (`pygaf.aquifers.Aq1dSemifiniteConf` attribute), 8
`is_1d()` (`pygaf.aquifers.Aq1dSemifiniteUnconf` attribute), 9
`is_1d()` (`pygaf.aquifers.Aq2dConf` attribute), 11
`is_1d()` (`pygaf.aquifers.Aq2dUnconf` attribute), 13
`is_1d()` (`pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf` attribute), 32
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky` attribute), 37
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf` attribute), 40
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf` attribute), 43
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky` attribute), 47
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky` attribute), 50
`is_1d()` (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf` attribute), 53
`is_1d()` (`pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf` attribute), 56
`is_1d()` (`pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf` attribute), 59
`is_1d()` (`pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky` attribute), 62
`is_1d()` (`pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf` attribute), 66
`is_1d()` (`pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf` attribute), 69
`is_1d()` (`pygaf.solutions.theis_1935.TheisWell.Aq2dConf` attribute), 73
`is_2d()` (`pygaf.aquifers.Aq1dFiniteConf` attribute), 4
`is_2d()` (`pygaf.aquifers.Aq1dFiniteUnconf` attribute), 6
`is_2d()` (`pygaf.aquifers.Aq1dSemifiniteConf` attribute), 8
`is_2d()` (`pygaf.aquifers.Aq1dSemifiniteUnconf` attribute), 9
`is_2d()` (`pygaf.aquifers.Aq2dConf` attribute), 11

is_2d (pygaf.aquifers.Aq2dLeaky attribute), 12
 is_2d (pygaf.aquifers.Aq2dUnconf attribute), 14
 is_2d (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 50
 is_2d (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 53
 is_2d (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 56
 is_2d (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 59
 is_2d (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 63
 is_2d (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 66
 is_2d (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf attribute), 69
 is_2d (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 73
 is_2d (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf attribute), 59
 is_2d (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky attribute), 63
 is_2d (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 66
 is_2d (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69
 is_2d (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
 is_circular (pygaf.basins.CircBasin attribute), 16
 is_circular (pygaf.basins.RectBasin attribute), 17
 is_circular (pygaf.grids.BasinGrid.RectBasin attribute), 20
 is_circular (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid attribute), 33
 is_confined (pygaf.aquifers.Aq1dFiniteConf attribute), 4
 is_confined (pygaf.aquifers.Aq1dFiniteUnconf attribute), 6
 is_confined (pygaf.aquifers.Aq1dSemifiniteConf attribute), 8
 is_confined (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9
 is_confined (pygaf.aquifers.Aq2dConf attribute), 11
 is_confined (pygaf.aquifers.Aq2dLeaky attribute), 12
 is_confined (pygaf.aquifers.Aq2dUnconf attribute), 14
 is_confined (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 32
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 37
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 43
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 43
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 47
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 50
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf attribute), 53
 is_confined (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 56
 is_confined (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf attribute), 59
 is_confined (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky attribute), 63
 is_confined (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 66
 is_confined (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf attribute), 69
 is_confined (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
 is_confined (pygaf.aquifers.Aq1dFiniteConf attribute), 4
 is_confined (pygaf.aquifers.Aq1dFiniteUnconf attribute), 6
 is_confined (pygaf.aquifers.Aq1dSemifiniteConf attribute), 8
 is_confined (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9
 is_confined (pygaf.aquifers.Aq2dConf attribute), 11
 is_confined (pygaf.aquifers.Aq2dLeaky attribute), 12
 is_confined (pygaf.aquifers.Aq2dUnconf attribute), 14
 is_confined (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 32
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 37
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 43
 is_confined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 43
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 47
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 50
 is_confined (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf attribute), 53
 is_confined (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 56
 is_confined (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf attribute), 59
 is_confined (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky attribute), 63
 is_confined (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 66
 is_confined (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf attribute), 69
 is_confined (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
 is_confined (pygaf.aquifers.Aq1dFiniteConf attribute), 4

tribute), 4
 is_heterogeneous (pygaf.aquifers.Aq1dFiniteUnconf attribute), 6
 is_heterogeneous (pygaf.aquifers.Aq1dSemifiniteConf attribute), 8
 is_heterogeneous (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9
 is_heterogeneous (pygaf.aquifers.Aq2dConf attribute), 11
 is_heterogeneous (pygaf.aquifers.Aq2dLeaky attribute), 12
 is_heterogeneous (pygaf.aquifers.Aq2dUnconf attribute), 14
 is_heterogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 32
 is_heterogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dLeaky attribute), 32
 is_heterogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dUnconf attribute), 32
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 37
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dUnconf attribute), 40
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dLeaky attribute), 40
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 43
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dLeaky attribute), 43
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 43
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 47
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dUnconf attribute), 47
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 50
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dUnconf attribute), 50
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dLeaky attribute), 53
 is_heterogeneous (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dUnconf attribute), 53
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 56
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dLeaky attribute), 56
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dUnconf attribute), 56
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dLeaky attribute), 59
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dUnconf attribute), 59
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dLeaky attribute), 63
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 63
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ2.Aq2dLeaky attribute), 66
 is_heterogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ2.Aq2dUnconf attribute), 66
 is_heterogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69
 is_heterogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dLeaky attribute), 69
 is_heterogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dUnconf attribute), 69
 is_heterogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
 is_heterogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dLeaky attribute), 73
 is_heterogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dUnconf attribute), 73
 is_homogeneous (pygaf.aquifers.Aq1dFiniteConf attribute), 5
 is_homogeneous (pygaf.aquifers.Aq1dFiniteUnconf attribute), 6
 is_homogeneous (pygaf.aquifers.Aq1dSemifiniteConf attribute), 8
 is_homogeneous (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9
 is_homogeneous (pygaf.aquifers.Aq2dConf attribute), 11
 is_homogeneous (pygaf.aquifers.Aq2dLeaky attribute), 12
 is_homogeneous (pygaf.aquifers.Aq2dUnconf attribute), 14
 is_homogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 32
 is_homogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dLeaky attribute), 32
 is_homogeneous (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dUnconf attribute), 32
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 37
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dUnconf attribute), 40
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dLeaky attribute), 40
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 43
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 47
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dUnconf attribute), 47
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 50
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dUnconf attribute), 50
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dLeaky attribute), 53
 is_homogeneous (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dUnconf attribute), 53
 is_homogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 56
 is_homogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dLeaky attribute), 56
 is_homogeneous (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dUnconf attribute), 56
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dLeaky attribute), 59
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dUnconf attribute), 59
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dLeaky attribute), 63
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 63
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ2.Aq2dLeaky attribute), 66
 is_homogeneous (pygaf.solutions.mine_flow.MineTransStripUnconfQ2.Aq2dUnconf attribute), 66
 is_homogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69
 is_homogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dLeaky attribute), 69
 is_homogeneous (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dUnconf attribute), 69
 is_homogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
 is_homogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dLeaky attribute), 73
 is_homogeneous (pygaf.solutions.theis_1935.TheisWell.Aq2dUnconf attribute), 73

`is_infinite(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dFinite attribute), 69`
`is_infinite(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dFinite attribute), 73`
`is_infinite(pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dFinite attribute), 47`
`is_infinite(pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dFinite attribute), 50`
`is_infinite(pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dFinite attribute), 53`
`is_infinite(pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dFinite attribute), 56`
`is_infinite(pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dFinite attribute), 59`
`is_infinite(pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dFinite attribute), 63`
`is_infinite(pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dFinite attribute), 66`
`is_infinite(pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69`
`is_infinite(pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73`
`is_leaky(pygaf.aquifers.Aq1dFiniteConf attribute), 5`
`is_leaky(pygaf.aquifers.Aq1dFiniteUnconf attribute), 6`
`is_leaky(pygaf.aquifers.Aq1dSemifiniteConf attribute), 8`
`is_leaky(pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9`
`is_leaky(pygaf.aquifers.Aq2dConf attribute), 11`
`is_leaky(pygaf.aquifers.Aq2dLeaky attribute), 12`
`is_leaky(pygaf.aquifers.Aq2dUnconf attribute), 14`
`is_leaky(pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2d attribute), 32`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2d attribute), 37`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2d attribute), 40`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2d attribute), 43`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2d attribute), 47`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2d attribute), 50`
`is_leaky(pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2d attribute), 53`
`is_leaky(pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2d attribute), 56`
`is_leaky(pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2d attribute), 59`
`is_leaky(pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2d attribute), 63`
`is_leaky(pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2d attribute), 66`
`is_leaky(pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69`
`is_leaky(pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73`
`is_radial(pygaf.aquifers.Aq1dFiniteConf attribute), 5`
`is_rectangular(pygaf.basins.CircBasin attribute), 16`
`is_rectangular(pygaf.basins.RectBasin attribute), 17`
`is_rectangular(pygaf.grids.BasinGrid.RectBasin attribute), 20`
`is_rectangular(pygaf.solutions.glover_1960.GloverRectBasinSteady attribute), 33`
`is_rectangular(pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69`
`is_rectangular(pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73`
`is_semi_finite(pygaf.aquifers.Aq1dFiniteConf attribute), 5`
`is_semi_finite(pygaf.aquifers.Aq1dFiniteUnconf attribute), 6`
`is_semi_finite(pygaf.aquifers.Aq1dSemifiniteConf attribute), 8`
`is_semi_finite(pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9`
`is_semi_finite(pygaf.aquifers.Aq2dConf attribute), 11`
`is_semi_finite(pygaf.aquifers.Aq2dLeaky attribute), 12`
`is_semi_finite(pygaf.aquifers.Aq2dUnconf attribute), 14`
`is_semi_finite(pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2d attribute), 32`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2d attribute), 38`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2d attribute), 40`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2d attribute), 43`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2d attribute), 47`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2d attribute), 50`
`is_semi_finite(pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2d attribute), 53`
`is_semi_finite(pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2d attribute), 56`
`is_semi_finite(pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2d attribute), 59`
`is_semi_finite(pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2d attribute), 63`
`is_semi_finite(pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2d attribute), 66`
`is_semi_finite(pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFinite attribute), 69`
`is_semi_finite(pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73`

- attribute), 70
- is_seminefinite (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
- is_steady (pygaf.wells.SteadyWell attribute), 27
- is_steady (pygaf.wells.TransientWell attribute), 29
- is_transient (pygaf.wells.SteadyWell attribute), 27
- is_transient (pygaf.wells.TransientWell attribute), 29
- is_unconfined (pygaf.aquifers.Aq1dFiniteConf attribute), 5
- is_unconfined (pygaf.aquifers.Aq1dFiniteUnconf attribute), 6
- is_unconfined (pygaf.aquifers.Aq1dSemifiniteConf attribute), 8
- is_unconfined (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 9
- is_unconfined (pygaf.aquifers.Aq2dConf attribute), 11
- is_unconfined (pygaf.aquifers.Aq2dLeaky attribute), 13
- is_unconfined (pygaf.aquifers.Aq2dUnconf attribute), 14
- is_unconfined (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 32
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 38
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 40
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 43
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 47
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 50
- is_unconfined (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf attribute), 53
- is_unconfined (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 56
- is_unconfined (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf attribute), 59
- is_unconfined (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky attribute), 63
- is_unconfined (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 66
- is_unconfined (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf attribute), 70
- is_unconfined (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 73
- K (pygaf.aquifers.Aq1dFiniteConf property), 4
- K (pygaf.aquifers.Aq1dFiniteUnconf property), 5
- K (pygaf.aquifers.Aq1dSemifiniteConf property), 7
- K (pygaf.aquifers.Aq1dSemifiniteUnconf property), 8
- K (pygaf.aquifers.Aq2dConf property), 10
- K (pygaf.aquifers.Aq2dLeaky property), 12
- K (pygaf.aquifers.Aq2dUnconf property), 13
- K (pygaf.aquifers.Aquifer attribute), 14
- K (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf property), 31
- K (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 37
- K (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf property), 40
- K (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf property), 43
- K (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
- K (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 49
- K (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf property), 52
- K (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf property), 55
- K (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf property), 59
- K (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
- K (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf property), 65
- K (pygaf.solutions.mine_flow.Steady1dConfFlow.Aq1dFiniteConf property), 69
- K (pygaf.solutions.theis_1935.TheisWell.Aq2dConf property), 73
- Kleak (pygaf.aquifers.Aq2dLeaky property), 12
- Kleak (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 37
- Kleak (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
- Kleak (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 49
- Kleak (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
- L (pygaf.aquifers.Aq1dFiniteConf attribute), 3
- L (pygaf.aquifers.Aq1dFiniteConf property), 4
- L (pygaf.aquifers.Aq1dFiniteUnconf attribute), 5
- L (pygaf.aquifers.Aq1dFiniteUnconf property), 5
- L (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf attribute), 68
- L (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf property), 69
- leakr () (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD method), 38
- lfac (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD property), 38
- lx (pygaf.basins.RectBasin attribute), 16
- lx (pygaf.basins.RectBasin property), 17
- lx (pygaf.grids.BasinGrid.RectBasin attribute), 19

[lx \(pygaf.grids.BasinGrid.RectBasin property\), 20](#)
[lx \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute\), 33](#)
[lx \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property\), 33](#)
[ly \(pygaf.basins.RectBasin attribute\), 16](#)
[ly \(pygaf.basins.RectBasin property\), 17](#)
[ly \(pygaf.grids.BasinGrid.RectBasin attribute\), 19](#)
[ly \(pygaf.grids.BasinGrid.RectBasin property\), 20](#)
[ly \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute\), 33](#)
[ly \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property\), 33](#)

M

[MineSteadyRadLeakyDD \(class in pygaf.solutions.mine_flow\), 36](#)
[MineSteadyRadLeakyDD.Aq2dLeaky \(class in pygaf.solutions.mine_flow\), 36](#)
[MineSteadyRadUnconfQ \(class in pygaf.solutions.mine_flow\), 39](#)
[MineSteadyRadUnconfQ.Aq2dUnconf \(class in pygaf.solutions.mine_flow\), 39](#)
[MineSteadyRadUnconfQ2 \(class in pygaf.solutions.mine_flow\), 42](#)
[MineSteadyRadUnconfQ2.Aq2dUnconf \(class in pygaf.solutions.mine_flow\), 42](#)
[MineSteadyStripLeakyDD \(class in pygaf.solutions.mine_flow\), 45](#)
[MineSteadyStripLeakyDD.Aq2dLeaky \(class in pygaf.solutions.mine_flow\), 45](#)
[MineSteadyStripLeakyQ \(class in pygaf.solutions.mine_flow\), 48](#)
[MineSteadyStripLeakyQ.Aq2dLeaky \(class in pygaf.solutions.mine_flow\), 49](#)
[MineSteadyStripUnconfQ \(class in pygaf.solutions.mine_flow\), 52](#)
[MineSteadyStripUnconfQ.Aq2dConf \(class in pygaf.solutions.mine_flow\), 52](#)
[MineTransRadConfDD \(class in pygaf.solutions.mine_flow\), 55](#)
[MineTransRadConfDD.Aq2dConf \(class in pygaf.solutions.mine_flow\), 55](#)
[MineTransStripConfQ \(class in pygaf.solutions.mine_flow\), 58](#)
[MineTransStripConfQ.Aq2dConf \(class in pygaf.solutions.mine_flow\), 58](#)
[MineTransStripLeakyQ \(class in pygaf.solutions.mine_flow\), 61](#)
[MineTransStripLeakyQ.Aq2dLeaky \(class in pygaf.solutions.mine_flow\), 61](#)
[MineTransStripUnconfQ \(class in pygaf.solutions.mine_flow\), 64](#)

[MineTransStripUnconfQ.Aq2dUnconf \(class in pygaf.solutions.mine_flow\), 65](#)
[module](#)
[pygaf.aquifers, 3](#)
[pygaf.basins, 15](#)
[pygaf.bcs, 18](#)
[pygaf.grids, 19](#)
[pygaf.solutions.glover_1960, 30](#)
[pygaf.solutions.mine_flow, 36](#)
[pygaf.solutions.steady_flow, 68](#)
[pygaf.solutions.theis_1935, 72](#)
[pygaf.stresses, 23](#)
[pygaf.utils, 24](#)
[pygaf.wells, 26](#)

N

[name \(pygaf.aquifers.Aquifer attribute\), 14](#)
[name \(pygaf.basins.CircBasin attribute\), 15](#)
[name \(pygaf.basins.RectBasin attribute\), 16](#)
[name \(pygaf.grids.BasinGrid.RectBasin attribute\), 19](#)
[name \(pygaf.grids.WellGrid.Well attribute\), 22](#)
[name \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute\), 33](#)
[name \(pygaf.solutions.theis_1935.TheisWell.WellGrid.Well attribute\), 74](#)
[name \(pygaf.wells.SteadyWell attribute\), 26](#)
[name \(pygaf.wells.TransientWell attribute\), 28](#)
[name \(pygaf.wells.Well attribute\), 30](#)
[npts \(pygaf.grids.BasinGrid property\), 21](#)
[npts \(pygaf.grids.WellGrid property\), 23](#)
[npts \(pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid property\), 34](#)
[npts \(pygaf.solutions.theis_1935.TheisWell.WellGrid property\), 75](#)

P

[pandas \(pygaf.wells.TransientWell attribute\), 29](#)
[periods \(pygaf.stresses.StressSeries attribute\), 23](#)
[periods \(pygaf.stresses.StressSeries property\), 24](#)
[periods \(pygaf.wells.TransientWell.StressSeries attribute\), 28](#)
[periods \(pygaf.wells.TransientWell.StressSeries property\), 28](#)
[pf \(pygaf.grids.WellGrid.Well attribute\), 22](#)
[pf \(pygaf.grids.WellGrid.Well property\), 22](#)
[pf \(pygaf.solutions.theis_1935.TheisWell.WellGrid.Well attribute\), 74](#)
[pf \(pygaf.solutions.theis_1935.TheisWell.WellGrid.Well property\), 74](#)
[pf \(pygaf.wells.SteadyWell attribute\), 26](#)
[pf \(pygaf.wells.SteadyWell property\), 27](#)
[pf \(pygaf.wells.TransientWell attribute\), 28](#)
[pf \(pygaf.wells.TransientWell property\), 29](#)
[pf \(pygaf.wells.Well attribute\), 30](#)

- pf (*pygaf.wells.Well* property), 30
 plot() (*pygaf.stresses.StressSeries* method), 24
 plot() (*pygaf.wells.TransientWell.StressSeries* method), 28
 pts (*pygaf.grids.BasinGrid* property), 21
 pts (*pygaf.grids.WellGrid* property), 23
 pts (*pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid* property), 34
 pts (*pygaf.solutions.theis_1935.TheisWell.WellGrid* property), 75
 pygaf.aquifers module, 3
 pygaf.basins module, 15
 pygaf.bcs module, 18
 pygaf.grids module, 19
 pygaf.solutions.glover_1960 module, 30
 pygaf.solutions.mine_flow module, 36
 pygaf.solutions.steady_flow module, 68
 pygaf.solutions.theis_1935 module, 72
 pygaf.stresses module, 23
 pygaf.utils module, 24
 pygaf.wells module, 26
- ## Q
- q (*pygaf.wells.SteadyWell* attribute), 26
 q() (*pygaf.solutions.steady_flow.SteadyIdConfFlow* method), 71
 qf (*pygaf.solutions.theis_1935.TheisWell* attribute), 72
 qfp (*pygaf.solutions.theis_1935.TheisWell* property), 76
 qp (*pygaf.solutions.mine_flow.MineSteadyRadLeakyDD* property), 39
 qp (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ* property), 41
 qp (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2* property), 45
 qp (*pygaf.solutions.mine_flow.MineSteadyStripLeakyDD* property), 48
 qp (*pygaf.solutions.mine_flow.MineSteadyStripLeakyQ* property), 51
 qp (*pygaf.solutions.mine_flow.MineSteadyStripUnconfQ* property), 54
 qp (*pygaf.solutions.mine_flow.MineTransRadConfDD* property), 58
 qp() (*pygaf.solutions.mine_flow.MineTransStripConfQ* method), 61
 qp() (*pygaf.solutions.mine_flow.MineTransStripLeakyQ* method), 64
 qp() (*pygaf.solutions.mine_flow.MineTransStripUnconfQ* method), 67
 qp() (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2* property), 45
 qp2 (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2* property), 45
 qp_cum() (*pygaf.solutions.mine_flow.MineTransStripConfQ* method), 61
 qp_cum() (*pygaf.solutions.mine_flow.MineTransStripUnconfQ* method), 67
 qp_steady (*pygaf.solutions.mine_flow.MineTransStripLeakyQ* property), 64
- ## R
- r (*pygaf.grids.WellGrid.Well* attribute), 22
 r (*pygaf.grids.WellGrid.Well* property), 22
 R (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ* property), 40
 R (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2* property), 44
 R (*pygaf.solutions.mine_flow.MineSteadyStripUnconfQ* property), 53
 R (*pygaf.solutions.steady_flow.SteadyIdConfFlow* attribute), 68
 r (*pygaf.solutions.theis_1935.TheisWell.WellGrid.Well* attribute), 74
 r (*pygaf.solutions.theis_1935.TheisWell.WellGrid.Well* property), 75
 r (*pygaf.wells.SteadyWell* attribute), 26
 r (*pygaf.wells.SteadyWell* property), 27
 r (*pygaf.wells.TransientWell* attribute), 27
 r (*pygaf.wells.TransientWell* property), 29
 r (*pygaf.wells.Well* attribute), 30
 r (*pygaf.wells.Well* property), 30
 rad (*pygaf.basins.CircBasin* property), 16
 RectBasin (class in *pygaf.basins*), 16
 ri (*pygaf.solutions.mine_flow.MineSteadyRadLeakyDD* property), 39
 ri (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ* property), 42
 ri (*pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2* property), 45
 ri() (*pygaf.solutions.mine_flow.MineTransRadConfDD* method), 58
 ri() (*pygaf.solutions.theis_1935.TheisWell* method), 76
 rinf() (*pygaf.solutions.theis_1935.TheisWell* method), 77
 rot (*pygaf.basins.CircBasin* attribute), 15
 rot (*pygaf.basins.RectBasin* attribute), 16
 rot (*pygaf.basins.RectBasin* property), 17

- rot (pygaf.grids.BasinGrid.RectBasin attribute), 19
 rot (pygaf.grids.BasinGrid.RectBasin property), 20
 rot (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin attribute), 33
 rot (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property), 34
 rot_rad (pygaf.basins.RectBasin property), 17
 rot_rad (pygaf.grids.BasinGrid.RectBasin property), 20
 rot_rad (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property), 34
 rotate_grid() (in module pygaf.utils), 25
 rotate_grid_2d() (in module pygaf.utils), 25
 rotate_point() (in module pygaf.utils), 26
 rp (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD property), 39
 rp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ property), 42
 rp (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2 property), 45
 rp (pygaf.solutions.mine_flow.MineTransRadConfDD property), 58
- ## S
- S (pygaf.aquifers.Aq1dFiniteConf property), 4
 S (pygaf.aquifers.Aq1dFiniteUnconf property), 5
 S (pygaf.aquifers.Aq1dSemifiniteConf property), 7
 S (pygaf.aquifers.Aq1dSemifiniteUnconf property), 8
 S (pygaf.aquifers.Aq2dConf property), 10
 S (pygaf.aquifers.Aq2dLeaky property), 12
 S (pygaf.aquifers.Aq2dUnconf property), 13
 S (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf property), 31
 S (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 37
 S (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf property), 40
 S (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf property), 43
 S (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
 S (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 50
 S (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf property), 53
 S (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf property), 55
 S (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf property), 59
 S (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
 S (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf property), 65
 S (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf property), 69
 S (pygaf.solutions.theis_1935.TheisWell.Aq2dConf property), 73
 series (pygaf.wells.TransientWell.StressSeries property), 24
 series (pygaf.wells.TransientWell.StressSeries property), 24
 Ss (pygaf.aquifers.Aq1dFiniteConf attribute), 3
 Ss (pygaf.aquifers.Aq1dFiniteConf property), 4
 Ss (pygaf.aquifers.Aq1dSemifiniteConf attribute), 7
 Ss (pygaf.aquifers.Aq1dSemifiniteConf property), 7
 Ss (pygaf.aquifers.Aq2dConf attribute), 10
 Ss (pygaf.aquifers.Aq2dConf property), 10
 Ss (pygaf.aquifers.Aq2dLeaky attribute), 11
 Ss (pygaf.aquifers.Aq2dLeaky property), 12
 Ss (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf attribute), 31
 Ss (pygaf.solutions.glover_1960.GloverRectBasinSteady.Aq2dConf property), 31
 Ss (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky attribute), 36
 Ss (pygaf.solutions.mine_flow.MineSteadyRadLeakyDD.Aq2dLeaky property), 37
 Ss (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky attribute), 46
 Ss (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD.Aq2dLeaky property), 46
 Ss (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky attribute), 49
 Ss (pygaf.solutions.mine_flow.MineSteadyStripLeakyQ.Aq2dLeaky property), 50
 Ss (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf attribute), 52
 Ss (pygaf.solutions.mine_flow.MineSteadyStripUnconfQ.Aq2dConf property), 53
 Ss (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf attribute), 55
 Ss (pygaf.solutions.mine_flow.MineTransRadConfDD.Aq2dConf property), 56
 Ss (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf attribute), 58
 Ss (pygaf.solutions.mine_flow.MineTransStripConfQ.Aq2dConf property), 59
 Ss (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky attribute), 61
 Ss (pygaf.solutions.mine_flow.MineTransStripLeakyQ.Aq2dLeaky property), 62
 Ss (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dLeaky attribute), 68
 Ss (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dLeaky property), 69
 Ss (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf attribute), 68
 Ss (pygaf.solutions.steady_flow.Steady1dConfFlow.Aq1dFiniteConf property), 69
 Ss (pygaf.solutions.theis_1935.TheisWell.Aq2dConf attribute), 72
 Ss (pygaf.solutions.theis_1935.TheisWell.Aq2dConf property), 73
 ss (pygaf.wells.TransientWell attribute), 27

- state (pygaf.wells.SteadyWell property), 27
 state (pygaf.wells.TransientWell property), 29
 Steady1dConfFlow (class in pygaf.solutions.steady_flow), 68
 Steady1dConfFlow.Aq1dFiniteConf (class in pygaf.solutions.steady_flow), 68
 Steady1dConfFlow.SteadyBC (class in pygaf.solutions.steady_flow), 70
 SteadyBC (class in pygaf.bcs), 18
 SteadyWell (class in pygaf.wells), 26
 StressSeries (class in pygaf.stresses), 23
 swl (pygaf.aquifers.Aq1dFiniteUnconf property), 6
 swl (pygaf.aquifers.Aq1dSemifiniteUnconf property), 9
 swl (pygaf.aquifers.Aq2dUnconf property), 14
 swl (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf property), 40
 swl (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf property), 43
 swl (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf property), 66
 Sy (pygaf.aquifers.Aq1dFiniteUnconf attribute), 5
 Sy (pygaf.aquifers.Aq1dFiniteUnconf property), 6
 Sy (pygaf.aquifers.Aq1dSemifiniteUnconf attribute), 8
 Sy (pygaf.aquifers.Aq1dSemifiniteUnconf property), 9
 Sy (pygaf.aquifers.Aq2dUnconf attribute), 13
 Sy (pygaf.aquifers.Aq2dUnconf property), 13
 Sy (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf attribute), 39
 Sy (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ.Aq2dUnconf property), 40
 Sy (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf attribute), 42
 Sy (pygaf.solutions.mine_flow.MineSteadyRadUnconfQ2.Aq2dUnconf property), 43
 Sy (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf attribute), 65
 Sy (pygaf.solutions.mine_flow.MineTransStripUnconfQ.Aq2dUnconf property), 65
- ## T
- T (pygaf.aquifers.Aq1dFiniteUnconf property), 6
 T (pygaf.aquifers.Aq1dSemifiniteUnconf property), 9
 T (pygaf.aquifers.Aquifer property), 14
 TheisWell (class in pygaf.solutions.theis_1935), 72
 TheisWell.Aq2dConf (class in pygaf.solutions.theis_1935), 72
 TheisWell.WellGrid (class in pygaf.solutions.theis_1935), 73
 TheisWell.WellGrid.Well (class in pygaf.solutions.theis_1935), 74
 top (pygaf.aquifers.Aquifer property), 15
 TransientWell (class in pygaf.wells), 27
 TransientWell.StressSeries (class in pygaf.wells), 28
- ## U
- u1() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 36
 u2() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 36
 u3() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 36
 u4() (pygaf.solutions.glover_1960.GloverRectBasinSteady method), 36
- ## V
- value (pygaf.bcs.SteadyBC property), 18
 value (pygaf.solutions.steady_flow.Steady1dConfFlow.SteadyBC property), 70
 values (pygaf.stresses.StressSeries attribute), 24
 values (pygaf.stresses.StressSeries property), 24
 values (pygaf.wells.TransientWell.StressSeries attribute), 28
 values (pygaf.wells.TransientWell.StressSeries property), 29
 verts (pygaf.basins.RectBasin property), 17
 verts (pygaf.grids.BasinGrid.RectBasin property), 20
 verts (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property), 34
 verts_rot (pygaf.basins.RectBasin property), 17
 verts_rot (pygaf.grids.BasinGrid.RectBasin property), 20
 verts_rot (pygaf.solutions.glover_1960.GloverRectBasinSteady.BasinGrid.RectBasin property), 34
- ## W
- Well (class in pygaf.wells), 29
 WellGrid (class in pygaf.grids), 21
 WellGrid.Well (class in pygaf.grids), 22
- ## X
- x (pygaf.grids.WellGrid.Well attribute), 22
 x (pygaf.solutions.theis_1935.TheisWell.WellGrid.Well attribute), 74
 x (pygaf.wells.SteadyWell attribute), 26
 x (pygaf.wells.TransientWell attribute), 27
 x (pygaf.wells.Well attribute), 29
 xi (pygaf.solutions.mine_flow.MineSteadyStripLeakyDD property), 48

`xi` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ`
property), 52

`xi` (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ`
property), 54

`xi()` (`pygaf.solutions.mine_flow.MineTransStripConfQ`
method), 61

`xi()` (`pygaf.solutions.mine_flow.MineTransStripUnconfQ`
method), 67

`xi_steady` (`pygaf.solutions.mine_flow.MineTransStripLeakyQ`
property), 64

Y

`y` (`pygaf.grids.WellGrid.Well` attribute), 22

`Y` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyDD`
property), 47

`Y` (`pygaf.solutions.mine_flow.MineSteadyStripLeakyQ`
property), 50

`Y` (`pygaf.solutions.mine_flow.MineSteadyStripUnconfQ`
property), 53

`Y` (`pygaf.solutions.mine_flow.MineTransStripConfQ` prop-
erty), 59

`Y` (`pygaf.solutions.mine_flow.MineTransStripLeakyQ`
property), 63

`Y` (`pygaf.solutions.mine_flow.MineTransStripUnconfQ`
property), 66

`y` (`pygaf.solutions.theis_1935.TheisWell.WellGrid.Well`
attribute), 74

`y` (`pygaf.wells.SteadyWell` attribute), 26

`y` (`pygaf.wells.TransientWell` attribute), 27

`y` (`pygaf.wells.Well` attribute), 29