orangery Documentation

Release 0.4.3

Michael Rahnis

Apr 05, 2019

Contents

1 Orangery **1**
- 1.1 Installation . . . 2
- 1.2 Examples . . . 2
- 1.3 License . . . 2
- 1.4 Documentation . . . 2

2 Orangery Manual **3**

3 orangery **5**
- 3.1 orangery Package . . . 5

4 Indices and tables **13**

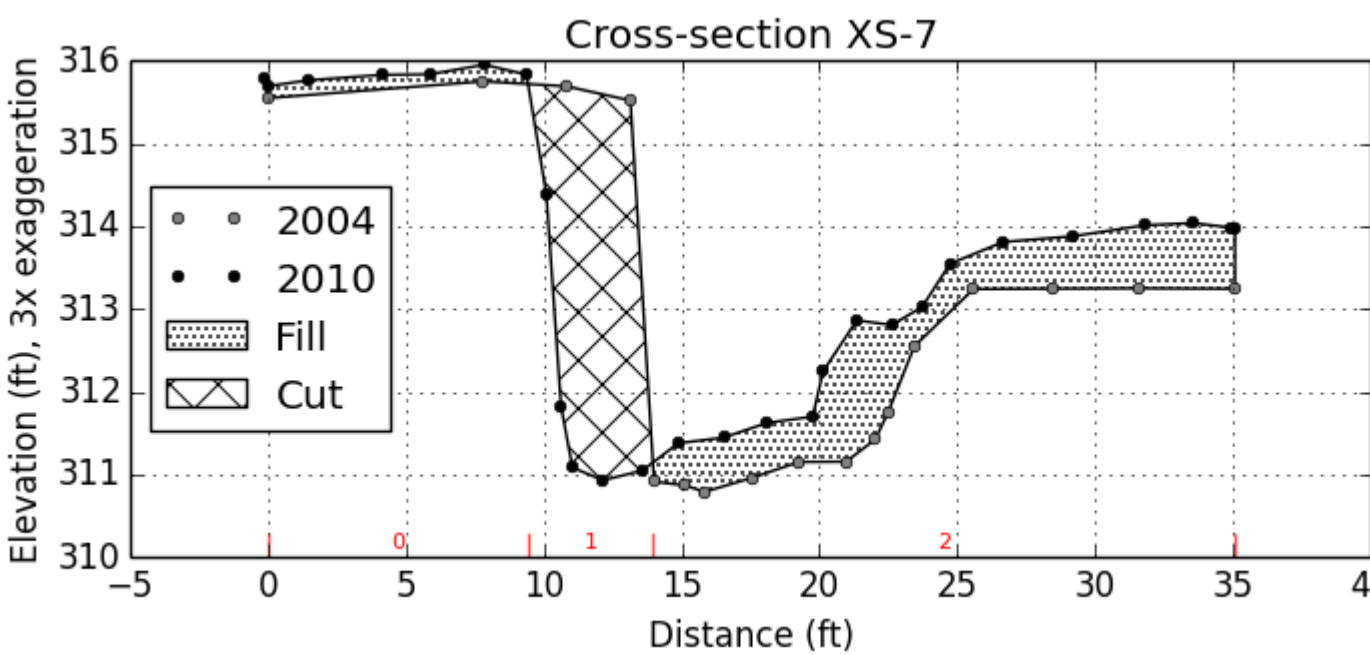
Python Module Index **15**

CHAPTER 1

Orangery

Orangery is a Python library to support analysis of topographic cross-sections, particularly on stream channels. The intent is to enable the user to write simple scripts that operate on CSV data exported from a survey data collector.

Orangery was initially a single script that allowed me to segregate, by grain size, changed areas on repeat topographic cross-sections. It can produce output plots like the one below.



1.1 Installation

To install from the Python Package Index:

$pip install orangery

To install from the source distribution execute the setup script in the orangery directory:

$python setup.py install

To install from Anaconda Cloud:

If you are starting from scratch the first thing to do is install the Anaconda Python distribution, add the necessary channels to obtain the dependencies and install orangery.

```
$conda config --append channels conda-forge
$conda config --append channels mrahnis
$conda install orangery
```

To install from the source distribution:

Execute the setup script in the orangery directory:

```
$python setup.py install
```

1.2 Examples

The example scripts may be run like so:

$python plots.py

1.3 License

BSD

1.4 Documentation

Latest html

CHAPTER 2

Orangery Manual

CHAPTER 3

orangery

3.1 orangery Package

3.1.1 `orangery` Package

3.1.2 `survey` Module

class `orangery.core.survey.`**`LevelSection`**(*data*, *p1*, *p2*, *backsight=0.0*, *datum=0.0*, *reverse=False*, *z_adjustment=None*)

Bases: `object`

A Section view of a set of d,z coordinates. Z values may be calculated based on backsight, foresight and datum elevation.

plot(*view='section'*, ***kwargs*)

Plot the d, z values of the data.

Parameters

- **view** (*str*) – Valid entries are 'section' and 'map'. Default is 'section' view.
- **kwargs** (*dict*) – Keyword arguments to be passed to Pandas and matplotlib.

Returns a matplotlib Axis.

Return type ax (Axis)

class `orangery.core.survey.`**`Section`**(*data*, *p1*, *p2*, *reverse=False*, *z_adjustment=None*)

Bases: `object`

A Section view of a set of x,y,z coordinates.

Parameters

- **data** (*pandas.DataFrame*) – contains the data to project.
- **p1** (*shapely.Point*) – the start of a line of section.

- **p2** (*shapely.Point*) – the end of a line of section.
- **reverse** (*bool*) – reverse the order of points in the section.
- **z_adjustment** (*float*) – adjust the elevation of the data.

plot (*view='section'*, ***kwargs*)
Plot the d, z values of the projected data.

Parameters

- **view** (*str*) – Valid entries are 'section' and 'map'. Default is 'section' view.
- **kwargs** (*dict*) – Keyword arguments to be passed to Pandas and matplotlib.

Returns a matplotlib Axis.

Return type ax (Axis)

class orangery.core.survey.**Survey** (*filename*, *columns*, *codebook*, *header=0*, ***kwargs*)
Bases: object

A Survey dataset.

Parameters

- **filename** (*str*) – the path to the file to read.
- **format** (*str*) – a string of characters that describes the survey data. Accepts: p - point x, e - x coord, easting y, n - y coord, northing z, h - z coord d, s - distance, station o - offset t - timestamp c - code r - remark q - quality f - foresight a - attribute The string may contain the characters in any order, without duplicates, except for 'a' which may occur multiple times. The string must contain x/e,y/n,z/h or d/s,z/h. Examples: 'pyxzctnff', 'pnezfrf'
- **codebook** (*dict*) – a dict that describes the codes used in the survey.
- **header** (*int*) – the row number of the header. As in pandas it is 0 by default. If there is no header row specify 'None'.
- **kwargs** (*dict*) – keyword arguments passed to pandas.read_csv.

plot (***kwargs*)
Plot the x, y values of the data.

Parameters **kwargs** (*dict*) – Keyword arguments to be passed to Pandas and matplotlib.

Returns a matplotlib Axis.

Return type ax (Axis)

save (*filename=None*, *original_header=False*, *write_history=False*)
Save the data to a file

translate (*deltas*)
Translate the data by an xyz offset and add a line to history.

3.1.3 evaluation Module

3.1.4 filter Module

orangery.core.filter.**benchmarks** (*df*, *code_table*, *codebook*)
Given a DataFrame return survey records with benchmark codes.

Parameters

- **df** (*pandas.DataFrame*) – survey data records.
- **code_table** (*pandas.DataFrame*) – survey data record properties extracted by parse function.
- **codebook** (*dict*) – a dict that describes the codes used in the survey.

Returns records having benchmark codes.

Return type result (DataFrame)

`orangery.core.filter.`**`controls`**(*df*, *code_table*, *codebook*)

Given a DataFrame return survey records that have control codes.

Parameters

- **df** (*pandas.DataFrame*) – survey data records.
- **code_table** (*pandas.DataFrame*) – survey data record properties extracted by parse function.
- **codebook** (*dict*) – a dict that describes the codes used in the survey.

Returns records having control codes.

Return type result (DataFrame)

`orangery.core.filter.`**`endpoints`**(*df*, *reverse=False*)

Given a DataFrame return the first and last survey records.

Parameters

- **df** (*pandas.DataFrame*) – survey data records.
- **reverse** (*bool*) – False returns first then last point, True returns last then first.

Returns first and last records in a DataFrame as Points.

Return type p1, p2 (Point)

`orangery.core.filter.`**`group`**(*df*, *code_table*, *group*, *exclude=[]*)

Given a DataFrame return a copy of the survey records belonging to a given group

Parameters

- **df** (*DataFrame*) – survey data records.
- **code_table** (*DataFrame*) – survey data record properties extracted by parse function.
- **group** (*str*) – name of the group to select.

Returns records matching the given group name.

Return type result (DataFrame)

`orangery.core.filter.`**`pointname`**(*df*, *name*)

Given a DataFrame return the named point or survey record.

Parameters

- **df** (*DataFrame*) – survey data records.
- **name** (*str*) – name of the point to select.

Returns records where point field (equivalent to the 'Point name' on Trimble data collectors) is equivalent to the name argument.

Return type result (DataFrame)

3.1.5 Subpackages

ops Package

geometry Module

orangery.ops.geometry.**close**(*line1*, *line2*)

orangery.ops.geometry.**cut_by_distance**(*line*, *distance*)

This line cutting function is from shapely recipes http://sgillies.net/blog/1040/shapely-recipes/

Parameters

- **line** (*LineString*) – the line to cut.
- **distance** (*float*) – distance from beginning of line to cutting point.

Returns array of cut line segments.

Return type segments (LineString array)

orangery.ops.geometry.**cut_by_distances**(*line*, *intersections*)

Cut a line at multiple points by calculating the distance of each point along the line. Uses the cut_by_distance function.

Parameters

- **line** (*LineString*) – the line to cut.
- **intersections** (*MultiPoint*) – a MultiPoint object containing cut points

Returns contains the line segments.

Return type segments (MultiLineString)

orangery.ops.geometry.**cut_by_point**(*line*, *pt*)

A cut function that divides a line and inserts points at the cut location.

Parameters

- **line** (*LineString*) – the line to cut.
- **pt** (*Point*) – a point on the line where the cut is to be made.

Returns array of cut line segments.

Return type segments (LineString array)

orangery.ops.geometry.**cut_by_points**(*line*, *intersections*)

Cut a line at multiple points by breaking the line and inserting each point. Uses the cut_by_point function.

Parameters

- **line** (*LineString*) – the line to cut.
- **intersections** (*MultiPoint*) – a MultiPoint object containing the cut points.

Returns contains the line segments.

Return type segments (MultiLineString)

orangery.ops.geometry.**difference**(*line1*, *line2*, *close_ends=False*)

Create polygons from two LineString objects.

Parameters

- **line1** (*LineString*) – a line representing the initial condition.

- **line2** (*LineString*) – a line representing the final condition.
- **close_ends** (*bool*) – option to close open line ends with vertical line segments.

Returns the intersections between the LineString objects. polygons (Polygon array) : the polygons between the lines. signs (int array) : contains values of +1 or -1 to identify polygons as cut or fill.

Return type intersections (Point array)

`orangery.ops.geometry.extend`(*line*, *pt*, *prepend*)

Extends a LineString by one Point, which may be prepended at the start of the LineString, or appended at the end.

Parameters

- **line** (*LineString*) – the line to extend.
- **pt** (*Point*) – the coordinate to extend to.
- **prepend** (*bool*) – if True then prepend, else append.

Returns the extended LineString.

Return type newline (LineString)

`orangery.ops.geometry.project`(*p1*, *p2*, *p3*)

Project a Point, p3 onto a line between Points p1 and p2.

Uses Shapely and GEOS functions, which set distance to zero for all negative distances.

Parameters

- **p1** (*Point*) – point at zero distance on line between p1 and p2.
- **p2** (*Point*) – endpoint of line.
- **p3** (*Point*) – the point to project.

Returns the projected Point, disctance along line, offset from line, and fractional distance along line.

Return type result (dict)

`orangery.ops.geometry.project2`(*p1*, *p2*, *p3*)

Project a Point, p3 onto a line intersecting Points p1 and p2.

Adapted from tutorial by Paul Bourke: http://paulbourke.net/geometry/pointline/ This projection function allows for points at negative distances.

Parameters

- **p1** (*Point*) – point at zero distance on line between p1 and p2.
- **p2** (*Point*) – endpoint on line.
- **p3** (*Point*) – the point to project.

Returns the projected Point, distance along line, offset from line, and fractional distance along line.

Return type result (dict)

`orangery.ops.geometry.project_points`(*points*, *p1*, *p2*)

Project multiple points onto a line through Points p1, p2.

Parameters

- **points** (*pandas.DataFrame*) – survey data to project.

- **p1** (*Point*) – point at zero distance on line between p1 and p2.
- **p2** (*Point*) – endpoint of line.

Returns DataFrame of projected points, including x, y, z, distance along line, offset from line, and fractional distance along line.

Return type result (DataFrame)

`orangery.ops.geometry.sign`(*line1*, *line2*)

Determine left-right orientation of a line relative to another

Iterates over points in two lines to identify line intersections at identical coordinates. At each intersection looks ahead and projects the next coordinate from line2 onto line1, and determines whether line2 is left or right of line1. Left offsets give a negative sign representing cut; right offsets give positive sign representing fill.

Parameters

- **line1** (*LineString*) – the line representing the initial condition.
- **line2** (*LineString*) – the line representing the final condition.

Returns members are positive or negative integer one.

Return type signs (int array)

`orangery.ops.geometry.snap_to_points`(*segments*, *intersections*)

Snap line segment endpoints to given points

Compare segment endpoints in a MultiLineString against points in a Point array to within a given precision; if the points match then update the segment endpoint with the coordinate given in the Point array.

Parameters

- **segments** (*MultiLineString*) – the line segments to snap.
- **intersections** (*Point array*) – the points to snap to.

Returns an updated MultiLineString.

Return type newline (MultiLineString)

`orangery.ops.geometry.update`(*line*, *pt*, *idx*)

Update a point within a LineString

Parameters

- **line** (*LineString*) – the line to update.
- **pt** (*Point*) – the new coordinate.
- **idx** (*int*) – the integer index of the vertex to update.

Returns the updated LineString.

Return type newline (LineString)

`text` Module

`orangery.ops.text.parse`(*points*, *codebook*)

Parses the codes in a DataFrame to extract information about points and chains of points.

Parameters

- **points** (*DataFrame*) – contains the survey data.

- **codebook** (*dict*) – a dict that describes the codes used in the survey.

Returns

Describes the points and chains of points. Column names match keys in the codes sub-dict, the added group column currently comes from the 'comment' field at each start command.

Return type df (DataFrame)

`correction` Module

`orangery.ops.correction.`**`get_offsets`**(*df*, *coords*)

Calulate the x,y,z offsets between a dataframe record, and an array of x,y,z coordinates.

`orangery.ops.correction.`**`translate`**(*df*, *offsets*)

Translate the x,y,z coordinates for records in a dataframe by an array of offsets.

tools Package

`plotting` Module

`orangery.tools.plotting.`**`annotate_plot`**(*self*, *ax=None*)

Add annotation to a plot to identify individual polygons.

Parameters **ax** (*Axis*) – matplotlib Axis to which to add annotation.

Returns patched matplotlib Axis.

Return type ax (Axis)

`orangery.tools.plotting.`**`get_scale_factor`**(*fig*, *ax*, *scale*, *axis='x'*)

Get the scale factor needed to obtain a desired scale in x-axis units per inch.

Parameters

- **fig** (*Figure*) – the figure to scale.
- **ax** (*Axis*) – the axis to scale.
- **scale** (*int or float*) – the desired output scale.

Returns the scale factor to apply to the figure size.

Return type scale_factor (float)

`orangery.tools.plotting.`**`polygon_plot`**(*self*, *ax=None*, *fill_ec='black'*, *fill_fc='none'*, *fill_hatch='...'*, *fill_label=None*, *cut_ec='black'*, *cut_fc='none'*, *cut_hatch='x'*, *cut_label=None*)

Adds two groups of polygon patches to a matplotlib Axis.

Fill and label first polygon of each type separately, otherwise make sequential call to fill.

Parameters

- **ax** (*Axis*) – matplotlib Axis to which to add polygon patches.
- **fill_ec** (*str*) – fill polygon edge color.
- **fill_fc** (*str*) – fill polygon face color.
- **fill_hatch** (*str*) – fill polygon hatch pattern.
- **fill_label** (*str*) – fill polygon label.

- **cut_ec** (*str*) – cut polygon edge color.
- **cut_fc** (*str*) – cut polygon face color.
- **cut_hatch** (*str*) – cut polygon hatch pattern.
- **cut_label** (*str*) – cut polygon label.

Returns patched matplotlib Axis.

Return type ax (Axis)

opus Module

CHAPTER 4

Indices and tables

- genindex
- search

Python Module Index

O

orangery.__init__, 5
orangery.core.filter, 6
orangery.core.survey, 5
orangery.ops.correction, 11
orangery.ops.geometry, 8
orangery.ops.text, 10
orangery.tools.plotting, 11

Index

A

annotate_plot() (*in module orangery.tools.plotting*), 11

B

benchmarks() (*in module orangery.core.filter*), 6

C

close() (*in module orangery.ops.geometry*), 8
controls() (*in module orangery.core.filter*), 7
cut_by_distance() (*in module orangery.ops.geometry*), 8
cut_by_distances() (*in module orangery.ops.geometry*), 8
cut_by_point() (*in module orangery.ops.geometry*), 8
cut_by_points() (*in module orangery.ops.geometry*), 8

D

difference() (*in module orangery.ops.geometry*), 8

E

endpoints() (*in module orangery.core.filter*), 7
extend() (*in module orangery.ops.geometry*), 9

G

get_offsets() (*in module orangery.ops.correction*), 11
get_scale_factor() (*in module orangery.tools.plotting*), 11
group() (*in module orangery.core.filter*), 7

L

LevelSection (*class in orangery.core.survey*), 5

O

orangery.__init__ (*module*), 5
orangery.core.filter (*module*), 6
orangery.core.survey (*module*), 5
orangery.ops.correction (*module*), 11
orangery.ops.geometry (*module*), 8
orangery.ops.text (*module*), 10
orangery.tools.plotting (*module*), 11

P

parse() (*in module orangery.ops.text*), 10
plot() (*orangery.core.survey.LevelSection method*), 5
plot() (*orangery.core.survey.Section method*), 6
plot() (*orangery.core.survey.Survey method*), 6
pointname() (*in module orangery.core.filter*), 7
polygon_plot() (*in module orangery.tools.plotting*), 11
project() (*in module orangery.ops.geometry*), 9
project2() (*in module orangery.ops.geometry*), 9
project_points() (*in module orangery.ops.geometry*), 9

S

save() (*orangery.core.survey.Survey method*), 6
Section (*class in orangery.core.survey*), 5
sign() (*in module orangery.ops.geometry*), 10
snap_to_points() (*in module orangery.ops.geometry*), 10
Survey (*class in orangery.core.survey*), 6

T

translate() (*in module orangery.ops.correction*), 11
translate() (*orangery.core.survey.Survey method*), 6

U

update() (*in module orangery.ops.geometry*), 10