
platelib Documentation

Release 0.1.2-alpha

Emil Dandanell Agerschou

Apr 27, 2018

CONTENTS

1	Introduction	1
1.1	Summary	1
1.2	Disclaimer	1
1.3	Read the docs	1
2	Installation	3
2.1	Prerequisites	3
2.2	Download	3
2.3	Install	3
2.4	Upgrade	3
2.5	Uninstall	3
3	Support	5
4	Cookbook	7
4.1	Reading in data	7
4.2	Accessing data	7
4.3	Plotting data	8
5	platelib	9
5.1	platelib package	9
6	Contribute	11
6.1	More tools	11
6.2	Building	11
6.3	Planned improvements	12
	Python Module Index	13
	Index	15

INTRODUCTION

1.1 Summary

platelib is an attempt to make common tasks when working with kinetic platereader, especially amyloid aggregation, data easy and compatible with Python.

1.2 Disclaimer

The state of this repository is one of very early development, the code is not elegant, there most likely are bugs so **Use with caution!**

1.3 Read the docs

A brief overview is given below, for more detailed information see the docs directory and subfolders, or even better have a look at the source :)

INSTALLATION

2.1 Prerequisites

- `python`
- `pip`
- `git` (optional)

2.2 Download

Either download the [source distribution](#) directly or use:

```
git https://github.com/edager/platelib
```

2.3 Install

Go to directory where the platelib source file is (`platelib/dist/` if `git` was used) and run the following in the terminal:

```
pip install platelib-X.X.tar.gz
```

Where `X.X` should be replaced by the version that was downloaded.

2.4 Upgrade

Go to directory where the platelib source file is (`platelib/dist/` if `git` was used) and run the following in the terminal:

```
pip install --upgrade platelib-X.X.tar.gz
```

Where `X.X` should be replaced by the version that was downloaded.

2.5 Uninstall

Simply go to the terminal and run:

```
pip uninstall platelib
```


SUPPORT

- **For questions related to usage:**
 - [Stack Overflow](#) using the tags `python` and `matplotlib`
- **For questions related to bugs or enhancements:**
 - [GitHub](#) using the issue system

4.1 Reading in data

The main functionality of `platelib` is the `read_plate` function that allows for reading in platereader data from kinetic experiments into a common framework namely into the `Plate_data` class.

It can be specified how many replicates per sample were prepared (default is 3):

```
p = read_plate('path/to/file', replicates=5)
```

NOTE that the number of replicates have to be the same for all samples!

It can be specified which direction the replicates were loaded onto the plate where 'hori' (horizontal) means towards increasing numbers and 'vert' is towards increasing letters (default is 'hori'):

```
p = read_plate('path/to/file', rep_direction='vert')
```

NOTE that the replicates have to be next to each other!

Data from Tecan platereaders can be read in as (default is 'bmg'):

```
p = read_plate('path/to/file', platereader='tecan')
```

NOTE that this functionality has not been fully tested yet!

As well as from BMG platereaders either where the data has prior been transposed `True` such that well data are in column format or in row format `False` (default is `True`):

```
p = read_plate('path/to/file', transposed=False)
```

The time unit can also be specified which as either 'seconds', 'minutes', 'hours', or 'days' will carry along into indexes if exported and to unit of x-axis if plotted (default is 'hours'):

```
p = read_plate('path/to/file', time_unit='days')
```

4.2 Accessing data

The `Plate_data` class allows for different ways of accessing the data

Through index:

```
p[1]
```

Through index slice:

```
p[:, :3]
```

Through well name:

```
p['B02']
```

Through list of well names:

```
p[['B02', 'C03', 'D04']]
```

Retrieved as a pandas.DataFrame with wellnames as column names and time points as index:

```
df = Plate_data.to_a_dataframe()
```

Or as a (C)omma (S)eperated (V)aribles file with the first line being (time unit +) well names and the first column are the time points:

```
Plate_data.to_a_csv('path/to/file.csv')
```

4.3 Plotting data

The data is plotted according to replicates, and subtitles can be added (default is *None*), and it can be specified whether all plots should have the same y-axis (default is *True*):

```
p.plot(titles=['condition 1', 'conditions 2'], sharey=False)
```

Will make the number of samples plots, all replicates of a given condition in the given plot.

5.1 platelib package

5.1.1 Submodules

5.1.2 platelib.fitfun module

`platelib.fitfun.exp_rise` (*t, a, b, k*)

class `platelib.fitfun.fit_plate` (*data, replicates=3, rep_direction='hori'*)

Bases: `platelib.plateread.Plate_data`

`platelib.fitfun.gauss` (*x, amp, cen, sigma*)

basic gaussian

`platelib.fitfun.gauss_dataset` (*params, i, x*)

calc gaussian from params for data set i using simple, hardwired naming convention

`platelib.fitfun.linear` (*t, a, b*)

`platelib.fitfun.objective` (*params, x, data*)

calculate total residual for fits to several data sets held in a 2-D array, and modeled by Gaussian functions

`platelib.fitfun.quadratic` (*t, a, b, c*)

`platelib.fitfun.sigmoidal_auto` (*t, a, b, k*)

5.1.3 platelib.plateread module

class `platelib.plateread.Plate_data` (*data, replicates=3, rep_direction='hori'*)

Class for containing data from a platereader assay.

param data A pandas DataFrame with time points as index and wells as columns

param replicates Positive integer of replicates, assuming equal number of replicates of all samples

plot_data (*titles=None, sharey=True*)

Plots the number the number of sample i.e. replicates/wells in the data set.

Parameters

- **titles** – List-like object of subtitles
- **sharey** – Boolean, default is True, where the y-axis limits are identical. If False y-axis limits are given by matplotlib defaults.

to_a_csv (*path*)

Returns the data as a pandas dataframe with times as indexes

Parameters **path** – path to store output at

to_a_dataframe ()

Returns the data as a pandas dataframe with times as indexes

`platelib.plateread.read_plate` (*filename*, *replicates=3*, *rep_direction='hori'*, *time_unit='hours'*,
platereader='bmg', *transposed=True*)

Reads in data from a CSV file from a BMG or Tecan platereader and returns a platedata object.

Parameters

- **filename** – Path to filename as a string
- **replicates** – The number of replicates per sample, excepts a positive integer.
- **rep_direction** – The directions replicates is in. Only 'hori' and 'vert' are accepted directions, 'hori' if replicates are going from left to righth 'vert' from replicates going from top to bottom.
- **time_unit** – The time unit one would like to have, accepted values are: 'seconds', 'minutes', 'hours', 'days'
- **platereader** – The plate reader used to collect the data. Only 'bmg' and 'tecan' are accepted platereaders
- **transposed** – Whether the wells are in column (True) or row format (False).

`platelib.plateread.read_tecan` (*filename*)

Reads in untransposed data from a tecan platereader and returns a pandas DataFrame object.

Parameters **filename** – Path to filename as a string

`platelib.plateread.read_transposed_bmg` (*filename*)

Reads in transposed data from a BMG platereader and returns a pandas DataFrame object.

Parameters **filename** – Path to filename as a string

`platelib.plateread.read_untransposed_bmg` (*filename*)

Reads in untransposed data from a BMG platereader and returns a pandas DataFrame object.

Parameters **filename** – Path to filename as a string

`platelib.plateread.vert_order` (*cols*, *reps*)

Helper function to reorder data if vertical replicates were made.

Parameters

- **cols** – List-like object of columns names (wells) all assumed to have the form 'A12'.
- **reps** – Positive integer number of replicates.

5.1.4 Module contents

CONTRIBUTE

Contributions are more than welcome, please raise an issue on the [github](#) page highlighting the bug/extension/compatibilities before doing a pull request.

6.1 More tools

Apart from the tools listed in [Installation](#) the following is needed:

- Unix-like system
- [git](#)
- [pandoc](#)

6.2 Building

You have made some wicked cool changes to the source code or the documentation that you want to share with the world, awesome!

Now there's just a few steps before they can be incorporated into the platelib master branch

6.2.1 Changing the version number

The versioning scheme of platelib should be done in reasonable accordance with the so called [Semantic versioning](#) where X.Y.Z should be read as MAJOR.MINOR.PATCH.

The version number has to be changed in to two files `setup.py` and `docs/source/conf.py`

6.2.2 Create new source distribution

Go to the docs folder and run:

```
./full_make.sh
```

If no errors occurred it can be uploaded to your local branch and a pull request can be made.

6.3 Planned improvements

This is as much a wish-list as literally planned improvements:

- Plotting
 - Plotting of data from several plates in some sensible way.
- Fitting
 - Local fitting of traces in plate
 - Global fitting of traces in plate
- Statistical analysis
 - Goodness-of-fit
 - Variance along traces, among replicates, and between conditions
- Python 3.X compatibility
- PyPI availability

PYTHON MODULE INDEX

p

platelib, [10](#)

platelib.fitfun, [9](#)

platelib.plateread, [9](#)

E

`exp_rise()` (in module `platelib.fitfun`), 9

F

`fit_plate` (class in `platelib.fitfun`), 9

G

`gauss()` (in module `platelib.fitfun`), 9

`gauss_dataset()` (in module `platelib.fitfun`), 9

L

`linear()` (in module `platelib.fitfun`), 9

O

`objective()` (in module `platelib.fitfun`), 9

P

`Plate_data` (class in `platelib.plateread`), 9

`platelib` (module), 10

`platelib.fitfun` (module), 9

`platelib.plateread` (module), 9

`plot_data()` (`platelib.plateread.Plate_data` method), 9

Q

`quadratic()` (in module `platelib.fitfun`), 9

R

`read_plate()` (in module `platelib.plateread`), 10

`read_tecan()` (in module `platelib.plateread`), 10

`read_transposed_bmg()` (in module `platelib.plateread`), 10

`read_untransposed_bmg()` (in module `platelib.plateread`),
10

S

`sigmoidal_auto()` (in module `platelib.fitfun`), 9

T

`to_a_csv()` (`platelib.plateread.Plate_data` method), 9

`to_a_dataframe()` (`platelib.plateread.Plate_data` method),
10

V

`vert_order()` (in module `platelib.plateread`), 10