phototdt Documentation

Release 0.0.1

Matias Andina

Apr 18, 2023

CONTENTS:

1 phototdt **1**
- 1.1 Features . . . 1
- 1.2 Credits . . . 1

2 Installation **3**
- 2.1 Stable release . . . 3
- 2.2 From sources . . . 3

3 Usage **5**

4 Contributing **7**
- 4.1 Types of Contributions . . . 7
- 4.2 Get Started! . . . 8
- 4.3 Pull Request Guidelines . . . 9
- 4.4 Tips . . . 9
- 4.5 Deploying . . . 9

5 Credits **11**
- 5.1 Development Lead . . . 11
- 5.2 Contributors . . . 11

6 History **13**
- 6.1 0.0.1 (2022-10-05) . . . 13

7 Indices and tables **15**

CHAPTER

ONE

PHOTOTDT

This Python package contains functions to get photometry data from a Tucker-Davis Technology (TDT) photomerty system and calculate dFF using methods developed by Martianova and colleagues. For more information on the analysis method, you can visit Martianova, E., Aronson, S., Proulx, C.D. Multi-Fiber Photometry to Record Neural Activity in Freely Moving Animal.. J. Vis. Exp. (152), e60278, doi: 10.3791/60278 (2019).. Implementation details and other language implementations (R, Matlab) are archived in the publication's repository

- Free software: BSD license
- Documentation: https://phototdt.readthedocs.io.

1.1 Features

This package reads TDT data from the directory of the block (e.g., `photometry_dir`)

- Use `photo_data = phototdt.get_tdt_data(photometry_dir)` to read and obtain a DataFrame photometry data.
- Use `phototdt.tdt_to_csv.tdt_to_csv(photometry_dir)` to convert block to a csv file and calculate zdFF on the 465 channel.
- Use `phototdt.get_cam_timestamps(photometry_dir)` to read camera timestamps from block.

1.2 Credits

This package was created with Cookiecutter and the audreyr/cookiecutter-pypackage project template.

CHAPTER

TWO

INSTALLATION

2.1 Stable release

To install phototdt, run this command in your terminal:

```
$ pip install phototdt
```

This is the preferred method to install phototdt, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

2.2 From sources

The sources for phototdt can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/matiasandina/phototdt
```

Or download the tarball:

```
$ curl -OJL https://github.com/matiasandina/phototdt/tarball/master
```

Once you have a copy of the source, you can install it with:

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

CHAPTER

THREE

USAGE

To use phototdt in a project:

```
import phototdt
session_folder = "path/to/block/folder"
# Get tdt data
photo_data = phototdt.get_tdt_data(session_folder)
# Get camera timestamps
from phototdt.phototdt import get_cam_timestamps
cam_timestamps = get_cam_timestamps(folder=session_folder)
# Convert photometry data from block to csv (interactive if folder is None)
from phototdt.tdt_to_csv import tdt_to_csv
tdt_to_csv(session_folder)
```

::

You can rename the block contents into BIDS format::

from phototdt.rename_block import rename_block session_folder = "path/to/block/folder" rename_block(session_folder)

::

CHAPTER

FOUR

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/matiasandina/phototdt/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

phototdt could always use more documentation, whether as part of the official phototdt docs, in docstrings, or even on the web in blog posts, articles, and such.

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/matiasandina/phototdt/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here's how to set up *phototdt* for local development.

1. Fork the *phototdt* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/phototdt.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv phototdt
$ cd phototdt/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 phototdt tests
$ python setup.py test or pytest
$ tox
```

   To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 3.5, 3.6, 3.7 and 3.8, and for PyPy. Check https://travis-ci.com/matiasandina/phototdt/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ pytest tests.test_phototdt
```

4.5 Deploying

A reminder for the maintainers on how to deploy. Make sure all your changes are committed (including an entry in HISTORY.rst). Then run:

```
$ bump2version patch # possible: major / minor / patch
$ git push
$ git push --tags
```

Travis will then deploy to PyPI if tests pass.

CHAPTER

FIVE

CREDITS

5.1 Development Lead

- Matias Andina <matiasandina@gmail.com>

5.2 Contributors

This package contains substantial amount of work from the following contributors:

- Ekaterina Martianova <ekaterina.martianova.1@ulaval.ca>
- Renato Lombardo <renato.lombardo@unipa.it>

CHAPTER

SIX

HISTORY

6.1 0.0.1 (2022-10-05)

- First github release.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search