Fourier-filtered Relative Entropy Minimization Documentation

Release 0.1.0

Carl Simon Adorf

Nov 27, 2018

Contents:

CHAPTER 1

Fourier-filtered Relative Entropy Minimization

Implementation of the Fourier-filtered Entropy Minimization (FF-REM) method for HOOMD-blue. The method is described in detail in the associated publication:

> Carl S. Adorf, James Antonaglia, Julia Dshemuchadse, Sharon C. Glotzer, 2018. DOI: 10.1063/1.5063802.

- Free software: MIT license
- Documentation: https://ff-rem.readthedocs.io.

1.1 Quickstart

A complete example for the recovery of a Lennard-Jones potential is shown in `examples/lennard-jones`.

1.2 Requirements

- numpy
- HOOMD-blue
- gsd

1.3 Testing

To execute unit tests, run:

```
$ python -m unittest discover tests/
```

within the package root directory.

1.4 Credits

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

CHAPTER 2

Installation

2.1 Stable release

To install Fourier-filtered Relative Entropy Minimization, run this command in your terminal:

```
$ pip install ff-rem
```

This is the preferred method to install Fourier-filtered Relative Entropy Minimization, 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 Fourier-filtered Relative Entropy Minimization can be downloaded from the **'Github repo'_**.

You can either clone the public repository:

```
$ git clone git://bitbucket.org/glotzer/ff-rem
```

Or download the tarball:

```
$ curl -Ol https://bitbucket.org/glotzer/ff-rem/get/master.zip
```

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

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

CHAPTER 3

Usage

To use Fourier-filtered Relative Entropy Minimization in a project:

```
import ffrem
```

CHAPTER 4

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://bitbucket.org/glotzer/ff-rem/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 Bitbucket 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 Bitbucket issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

4.1.4 Write Documentation

Fourier-filtered Relative Entropy Minimization could always use more documentation, whether as part of the official Fourier-filtered Relative Entropy Minimization 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://bitbucket.org/glotzer/ff-rem/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 *ffrem* for local development.

1. Fork the *ffrem* repo on Bitbucket.
2. Clone your fork locally:

```
$ git clone git@bitbucket.org:your_name_here/ffrem.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 ffrem
$ cd ffrem/
$ 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 ffrem tests
$ python setup.py test or py.test
$ tox
```

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

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

```
$ 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 Bitbucket 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 3.4, 3.5 and 3.6, and for PyPy.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_ffrem
```

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:

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

Travis will then deploy to PyPI if tests pass.

CHAPTER 5

Credits

5.1 Development Lead

- Carl Simon Adorf <csadorf@umich.edu>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

6.1 0.1.0 (2018-10-17)

- First release on PyPI.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search