langevin$_{d}ynamics Documentation$

Release 0.1.0

xinyang li

December 14, 2016

Contents

Contents:

CHAPTER 1

langevin_dynamics

In statistical physics, a Langevin equation (Paul Langevin, 1908) is a stochastic differential equation describing the time evolution of a subset of the degrees of freedom. Python Boilerplate contains all the boilerplate you need to create a Python package.

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

1.1 Features

- A simple python program for 2D many-particle Lagevin equation simulation.
- Required input values are read from a file named input and output file is called trajectory.txt.
- Potential is based on simply y = c*sin (a*x^2+ b*y^2), which may not be physical at all. You can change a,b and c in main program to get your own potential file.
- Periodic boundary conditions enabled.
- Paralleled main dynamics loop
- Real-time display is added to the program. (Note: cause the program to become really slow.)
- For more information please check langevin_dynamics.info.

1.2 Note

- Please modify input under langevin_dynamcis folder before running simulations.

1.3 TODO

- Adding a module to convert tracjectories into gif to avoid performance issue.
- Including more physical potentials, such as Lennard-Jones potential.
- Re-structure the code to use higher level parallelism, and may introduce C/Fortran implementation for heavy computations.

1.4 Credits

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

This folder contains simply the documentations for lagevin dynamics code.

CHAPTER 2

Installation

2.1 Stable release

To install langevin_dynamics, run this command in your terminal:

```
$ pip install langevin_dynamics
```

This is the preferred method to install langevin_dynamics, 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 langevin_dynamics can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/tautomer/langevin_dynamics
```

Or download the tarball:

```
$ curl  -OL https://github.com/tautomer/langevin_dynamics/tarball/master
```

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

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

CHAPTER 3

Usage

To use langevin_dynamics in a project:

```
import langevin_dynamics
```

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://github.com/tautomer/langevin_dynamics/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

langevin_dynamics could always use more documentation, whether as part of the official langevin_dynamics 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/tautomer/langevin_dynamics/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 *langevin_dynamics* for local development.

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

```
$ git clone git@github.com:your_name_here/langevin_dynamics.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 langevin_dynamics
$ cd langevin_dynamics/
$ 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 langevin_dynamics 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 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 2.6, 2.7, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/tautomer/langevin_dynamics/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python3 -m unittest tests.test_langevin_dynamics
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search