MICS

Release 0.2.0

May 09, 2018

Contents

CHAPTER 1

Overview

Mixtures of Independently Collected Samples

- Free software: MIT license

1.1 Installation

```
pip install mics
```

1.2 Documentation

https://mics.readthedocs.io/

1.3 Development

To run the all tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

Windows	`set PYTEST_ADDOPTS=--cov-append` `tox`
Other	`PYTEST_ADDOPTS=--cov-append tox`

CHAPTER 2

Installation

At the command line:

```
pip install mics
```

CHAPTER 3

Usage

To use MICS in a project:

```
import mics
```

CHAPTER 4

Reference

4.1 mics

4.2 mixtures

class mics.mixtures.**mixture**

A mixture of independently collected samples (MICS)

Parameters

- **samples** (*list or tuple*) – a list of samples.
- **title** (*str, optional*) – a title.
- **verbose** (*bool, optional*) – a verbosity tag.
- **tol** (*float, optional*) – a tolerance.

free_energies(*reference=0*)

Returns a data frame containing the relative free energies of the datasetd samples of a *mixture*, as well as their standard errors.

reweighting(*potential*, *properties={}*, *derivatives={}*, *combinations={}*, *conditions=Empty DataFrame Columns: [] Index: []*, *reference=0*, ***kwargs*)

Performs reweighting of the properties computed by *functions* from the mixture to the samples determined by the provided *potential* with all *parameter* values.

Parameters

- **potential** (*string*)
- **properties** (*dict of strings*)
- **combinations** (*dict of strings*)
- **derivatives** (*dict of tuples*)
- **conditions** (*pandas.DataFrame*)

- **verbose** (*boolean*)
- ****kwargs**

4.3 samples

class mics.samples.**pool**(*label=''*, *verbose=False*)
A pool of independently collected samples.

class mics.samples.**sample**(*dataset*, *potential*, *autocorr=None*, *label=None*, *batchsize=None*, *verbose=False*, ***kwargs*)
A sample of configurations collected at a specific equilibrium state, aimed to be part of a mixture of independently collected samples (MICS).

Args:

dataset (pandas.DataFrame): a data frame whose rows represent configurations datasetd according to a given probability distribution and whose columns contain a number of properties evaluated for such configurations.

potential (function): the reduced potential that defines the equilibrium sample. This function might for instance receive **x** and return the result of an element-wise calculation involving **x["a"]**, **x["b"]**, etc, with **"a"**, **"b"**, etc being names of properties in **dataset**.

autocorr (function, optional): a function similar to **potential**, but whose result is an autocorrelated property to be used for determining the effective dataset size. If omitted, **potential** will be used to for this purpose.

Note: Formally, functions **potential** and **autocorr** must receive **x** and return **y**, where *length(y) == nrow(x)*.

4.4 utils

mics.utils.**covariance**(*y*, *ym*, *b*)
Computes the covariance matrix of the rows of matrix *y* among themselves. The method of Overlap Batch Mean (OBM) is employed with blocks of size *b*.

mics.utils.**cross_covariance**(*y*, *ym*, *z*, *zm*, *b*)
Computes the cross-covariance matrix between the rows of matrix *y* with those of matrix *z*. The method of Overlap Batch Mean (OBM) is employed with blocks of size *b*.

mics.utils.**genfunc**(*function*, *variables*, *constants*)
Returns a function based on the passed argument.

mics.utils.**multimap**(*functions*, *sample*)
Applies a list of `functions` to DataFrame *sample* and returns a numpy matrix whose number of rows is equal to the length of list *functions* and whose number of columns is equal to the number of rows in *sample*.

Note: Each function of the array might for instance receive *x* and return the result of an element-wise calculation involving *x["A"]*, *x["B"]*, etc, with "A", "B", etc being names of properties in DataFrame *sample*.

---

mics.utils.**overlapSampling**(*u*)
Computes the relative free energies of all sampled states using the Overlap Sampling method of Lee and Scott (1980).

mics.utils.**pinv**(*A*)

Computes the Moore-Penrose pseudoinverse of a symmetric matrix using eigenvalue decomposition.

CHAPTER 5

Contributing

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

5.1 Bug reports

When 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.

5.2 Documentation improvements

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

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/craabreu/mics/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 code contributions are welcome :)

5.4 Development

To set up *mics* for local development:

1. Fork mics (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/mics.git
```

3. Create a branch for local development:

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

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. 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
```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 6

Authors

- Charlles R. A. Abreu - http://atoms.peq.coppe.ufrj.br

CHAPTER 7

Changelog

7.1 0.2.0 (2018-05-09)

- Implementation of classes `sample`, `pool`, `mixture`, `MICS` and `MBAR`.

7.2 0.1.0 (2017-10-11)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

s

u

Index