alchemtest Documentation

Release 0.8.0+0.ge9d5401.dirty

David Dotson

Dec 09, 2022

OVERVIEW

alchemtest is a collection of test datasets for alchemical free energy calculations. The datasets come from a variety of software packages, primarily molecular dynamics engines, and are used as the test set for alchemlyb. The package is standalone, however, and can be used for any purpose.

Datasets are released under an open license that conforms to the Open Definition 2.1 that allows free use, re-use, redistribution, modification, separation, for any purpose and without a charge. All data and code can be found in the public GitHub repository alchemistry/alchemtest.

This library is **under active development**. We use semantic versioning to indicate clearly what kind of changes you may expect between releases. Although it is heavily used for the alchemlyb test suite it may contain bugs. Please raise any issues or questions in the Issue Tracker.

Note: *Contributions of data sets* are very welcome. Please raise an issue in the Issue Tracker to propose a new data set and we will help you with the process of adding it to **alchemtest**.

With release 0.7.0, the alchemlyb project adopted NEP 29 to determine which versions of Python and NumPy will be supported. When we release a new major or minor version, alchemtest will support *at least all minor versions of Python introduced and released in the prior 42 months from the release date with a minimum of 2 minor versions of Python*, and *all minor versions of NumPy released in the prior 24 months from the anticipated release date with a minimum of 3 minor versions of NumPy*.

CHAPTER

ONE

INSTALLING ALCHEMTEST

alchemtest is pure-Python, so it can be installed easily via `pip`:

```
pip install alchemtest
```

If you wish to install this in your user `site-packages`, use the `--user` flag:

```
pip install --user alchemtest
```

1.1 Installing from source

from source. Clone the source from GitHub with:

```
git clone https://github.com/alchemistry/alchemtest.git
```

then do:

```
cd alchemtest
pip install .
```

If you wish to install this in your user `site-packages`, use the `--user` flag:

```
pip install --user .
```

CHAPTER

TWO

BASIC USAGE

All datasets in `alchemtest` are accessible via `load_*` functions, organized in submodules by the software package that generated them. The current set of submodules are:

gmx	Gromacs molecular dynamics simulation datasets.
amber	Amber molecular dynamics simulation datasets.
namd	NAMD molecular dynamics simulation datasets.
gomc	GOMC Monte Carlo simulation datasets.

As an example, we can access the *Gromacs: Benzene in water* dataset with:

```
>>> from alchemtest.gmx import load_benzene
>>> bz = load_benzene()
```

and use the resulting *Bunch* object to introspect what this dataset includes. In particular, it features a `DESCR` attribute with a human-readable description of the dataset:

```
>>> print(bz.DESCR)
Gromacs: Benzene in water
=========================

Benzene in water, alchemically turned into benzene in vacuum separated from water

Notes
-----
Data Set Characteristics:
    :Number of Legs: 2 (Coulomb, VDW)
    :Number of Windows: 5 for Coulomb, 16 for VDW
    :Length of Windows: 40ns

    :Missing Values: None
    :Creator: \I. Kenney
    :Donor: Ian Kenney (ian.kenney@asu.edu)
    :Date: March 2017
    :License: `CC0
              <https://creativecommons.org/publicdomain/zero/1.0/>`_
              Public Domain Dedication

This dataset was generated using `MDPOW <https://github.com/Becksteinlab/MDPOW>`_,␣
↪with
the `Gromacs <http://www.gromacs.org/>`_ molecular dynamics engine.
```

as well as the dataset itself:

```
>>>  bz.data.keys()
['VDW', 'Coulomb']
```

which consists in this case of two alchemical legs, each having several files. For this dataset each file happens to correspond to a simulation sampling a particular λ:

```
>>> bz.data['Coulomb']
['/usr/local/python3.6/site-packages/alchemtest/gmx/benzene/Coulomb/0000/dhdl.xvg.bz2
↪',
 '/usr/local/python3.6/site-packages/alchemtest/gmx/benzene/Coulomb/0250/dhdl.xvg.bz2
↪',
 '/usr/local/python3.6/site-packages/alchemtest/gmx/benzene/Coulomb/0500/dhdl.xvg.bz2
↪',
 '/usr/local/python3.6/site-packages/alchemtest/gmx/benzene/Coulomb/0750/dhdl.xvg.bz2
↪',
 '/usr/local/python3.6/site-packages/alchemtest/gmx/benzene/Coulomb/1000/dhdl.xvg.bz2
↪']
```

These paths can be read by any appropriate parser for further analysis. For this particular dataset, see `alchemlyb.parsing.gmx` for a good set of parsers.

CHAPTER

THREE

CONTRIBUTING NEW DATA SETS

We are looking for new data sets. Please read the following and consider contributing data; details are described under *Process*.

3.1 Types of systems

The ideal set of files would be something like the GROMACS dataset for alchemtest `alchemtest.gmx`: benzene in water for 1...10 ns per window, with $\partial H/\partial\lambda$ saved every 10 ps. For GROMACS we tend to put each lambda in a separate directory (see the directory layout in alchemtest/gmx/benzene) but you should provide files that are typical of how the specific code is run.

3.2 Data set description

For your data set, you should be able to include the following in a brief description (which will become part of the data set and the documentation as described in more detail in *Process*):

- Include the value(s) that you get when analyzing the data set yourself so that we know the ground truth.

  It is very helpful if you include a brief explanation of how you analyze the data with alchemlyb or your own tool (show Python commands or the full command with options so that one can reproduce the analysis if necessary).
- State how the data set was generated. Include the temperature.
- Comment on what to look out for in the output files, e.g., special sampling options.
- For **new file formats:** Include information about the file format definition, such as *links* or *paper citations*.

In general, follow the example of the existing data sets (especially similar data sets or for the same MD/MC code) and discuss the specfics on an initial Pull Request.

3.3 Licensing

Finally, because we want to make the data part of the actual tests that are run every time when new code is committed to the repository, we would need the data to be made available under an open license (preferrable CC0 (public domain) or CC-BY (attribution required)). The dataset will carry the license and your authorship.

At the moment, all included data sets are in the public domain via CC0.

3.4 Process

1. Raise an issue in the alchemtest issue tracker proposing the new data set. In this issue we will do all discussions.
2. Fork the alchemtest repo and create a branch for your dataset.
3. Add your dataset to your branch. Follow the existing layout.
   - Choose a top level directory. If your data files are for GROMACS, add it to alchemtest/gmx or for NAMD to alchemtest/namd, etc. If you support a new code, create a new directory.
   - Create a *subdirectory* for your dataset, choose a good, short name for the dataset and the directory.
     - Create one or more additional directories inside your dataset directory for your actual data files; do whatever seems natural for your problem.
     - *Copy your data files to the appropriate subdirectories*. Consider compressing them with **`gzip`** or **`bzip2`** (alchemlyb can read compressed files).
     - Check the `MANIFEST.in`: make sure that the line

       ```
       recursive-include src/alchemtest *.gz *.bz2 *.zip *.rst *.txt *.out *.xvg
       ```

       will include your files into the package: If your filename extension(s) are not matched, add them.
     - Create a restructured text (reST) file `descr.rst` that describes the dataset. Look at other description files as examples: copy one that is close in what you need and modify. The description will show up in the online documentation and will be part of the dataset *`Bunch`*.
   - Add an accessor function `load_MYDATASET()` to the `access.py` file at the top of the code directory. The accessor function makes the dataset available as a `dict` under the *data* key in the *`Bunch`*. The data are typically another `dict` with different parts of a calculation such as Coulomb and VDW parts being different keys in a dictionary. All files that are needed for a single free energy calculation are in a `list` under the appropriate key. The description text is added the *DESCR* key.

     Again, copy an existing function and modify.
   - Add an `from .access import load_MYDATASET` to the top-level `__init__.py` to make your accessor function part of alchemtest.
4. Locally test that you can load your dataset:

   ```
   from alchemtest.MYCODE.MYDATASET import load_MYDATASET
   d = load_MYDATASET()
   print(d.DESCR)
   print(d.data)
   ```

   You should see your description and the full path to your datafiles (possibly inside another dictionary). It should be possible to work with your dataset as shown under *Basic usage*.

   Try building the documentation with

   ```
   python setup.py build_sphinx
   ```

   and look at the docs in `build/sphinx/html/index.html`.

   Check that your documentation is visible. If not, it's possible that another page needs to be added to the docs — just move ahead with the next step and ask in the comments on your Pull Request and we will help.
5. Create a Pull Request with your new code and files.

6. Add a *test* that checks that your files can be found. Look in the `src/alchemtest/tests` directory and follow the examples that are already there. We are also happy to help you with this step — just ask.

   You can run the tests locally with *pytest* and you will also see that the tests are run on your PR.

7. Engage in the code review — we might have questions, suggestions, and requests for revisions to ensure that your contribution fits into the library.

8. Once your PR is accepted it will be merged by a developer and your dataset is part of **alchemtest** — Congratulations!

CHAPTER

FOUR

HELPER FUNCTIONS AND CLASSES

A small number of functions and classes are included to help organize the data.

class alchemtest.**Bunch**(***kwargs*)

Container object for datasets

Dictionary-like object that exposes its keys as attributes.

```
>>> b = Bunch(a=1, b=2)
>>> b['b']
2
>>> b.b
2
>>> b.a = 3
>>> b['a']
3
>>> b.c = 6
>>> b['c']
6
```

Code taken from sklearn/utils/__init__.py version 0.19.1 under the 'New BSD license' https://github.com/scikit-learn/scikit-learn/blob/master/COPYING

CHAPTER

FIVE

GROMACS DATASETS

Gromacs molecular dynamics simulation datasets.

The *alchemtest.gmx* module features datasets generated using the Gromacs molecular dynamics engine. They can be accessed using the following accessor functions:

load_benzene()	Load the Gromacs benzene dataset.
load_ABFE()	Load the Gromacs ABFE dataset.
load_expanded_ensemble_case_1()	Load the Gromacs Host CB7 Guest C3 expanded ensemble dataset, case 1 (single simulation visits all states).
load_expanded_ensemble_case_2()	Load the Gromacs Host CB7 Guest C3 expanded ensemble dataset, case 2 (two simulations visit all states independently).
load_expanded_ensemble_case_3()	Load the Gromacs Host CB7 Guest C3 REX dataset, case 3.
load_water_particle_with_total_energy()	Load the Gromacs water particle with total energy dataset.
load_water_particle_with_potential_energy()	Load the Gromacs water particle with potential energy dataset.
load_water_particle_without_energy()	Load the Gromacs water particle without energy dataset.

5.1 Simple TI and FEP

The data sets contain derivatives of the Hamiltonian (TI) and free energy perturbation (FEP) data suitable for processing with FEP estimators as well as BAR/MBAR. Individual λ windows were run independently.

5.1.1 Gromacs: Benzene in water

Benzene in water, alchemically turned into benzene in vacuum separated from water

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 5 for Coulomb, 16 for VDW

Length of Windows 40ns

System Size 1668 atoms

Temperature 300 K

Pressure 1 bar

Alchemical Pathway vdw + coul –> vdw –> vacuum

Experimental Hydration Free Energy -0.90 +- 0.2 kcal/mol

Missing Values None

Energy unit kJ/mol

Time unit ps

Creator I. Kenney

Donor Ian Kenney (ian.kenney@asu.edu)

Date March 2017

License CC0 Public Domain Dedication

This dataset was generated using MDPOW, with the Gromacs molecular dynamics engine.

Experimental value sourced from [Mobley2013].

`alchemtest.gmx.load_benzene()`

Load the Gromacs benzene dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

5.2 Extended ensemble

Data for *extended ensemble* simulations; case 1 and case 2 are extended ensembles in the alchemical parameters, case 3 includes replica exchange (REX).

5.2.1 Gromacs: Host CB7 and Guest C3 in water

Host CB7 and Guest C3 in water, Guest C3 alchemically turned into Guest C3 in vacuum separated from water and Host CB7. This unpublished data uses Host CB7 and Guest C3 from [Muddana2014a]. Similar published data can be found in [Monroe2014a].

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 32 total, 20 for Coulomb, 12 for VDW

Number of Simulations 1

Length of Simulation 100ns

System Size 8286 atoms

Temperature 300 K

Alchemical Pathway vdw + coul –> vdw –> vacuum

Missing Values None

Energy unit kJ/mol

Time unit ps

Creator T. Jensen

Donor Travis Jensen (travis.jensen@colorado.edu)

Date November 2017

License CC0 Public Domain Dedication

This dataset was generated using the expanded ensemble algorithm in the Gromacs molecular dynamics engine.

`alchemtest.gmx.load_expanded_ensemble_case_1()`

Load the Gromacs Host CB7 Guest C3 expanded ensemble dataset, case 1 (single simulation visits all states).

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

5.2.2 Gromacs: Host CB7 and Guest C3 in water

Host CB7 and Guest C3 in water, Guest C3 alchemically turned into Guest C3 in vacuum separated from water and Host CB7. This unpublished data uses Host CB7 and Guest C3 from [Muddana2014b]. Similar published data can be found in [Monroe2014b].

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 32 total, 20 for Coulomb, 12 for VDW

Number of Simulations 2

Length of Simulation 50ns

System Size 8286 atoms

Temperature 300 K

Alchemical Pathway vdw + coul –> vdw –> vacuum

Missing Values None

Energy unit kJ/mol

Time unit ps

Creator T. Jensen

Donor Travis Jensen (travis.jensen@colorado.edu)

Date November 2017

License CC0 Public Domain Dedication

This dataset was generated using the expanded ensemble algorithm in the Gromacs molecular dynamics engine.

`alchemtest.gmx.load_expanded_ensemble_case_2()`

Load the Gromacs Host CB7 Guest C3 expanded ensemble dataset, case 2 (two simulations visit all states independently).

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

5.2.3 Gromacs: Host CB7 and Guest C3 in water

Host CB7 and Guest C3 in water, Guest C3 alchemically turned into Guest C3 in vacuum separated from water and Host CB7. This unpublished data uses Host CB7 and Guest C3 from [Muddana2014c].

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 32 total, 20 for Coulomb, 12 for VDW

Number of Simulations 32

Length of Simulation 5ns

System Size 8286 atoms

Temperature 300 K

Alchemical Pathway vdw + coul –> vdw –> vacuum

Missing Values None

Energy unit kJ/mol

Time unit ps

Creator T. Jensen

Donor Travis Jensen (travis.jensen@colorado.edu)

Date November 2017

License CC0 Public Domain Dedication

This dataset was generated using the REX algorithm in the Gromacs molecular dynamics engine.

`alchemtest.gmx.load_expanded_ensemble_case_3()`
Load the Gromacs Host CB7 Guest C3 REX dataset, case 3.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

5.3 Water particle TI and FEP

3 simple dH/dl and U_nk datasets of a single water particle from a simulations of water between to hydrophilic surfaces. One dataset contains a total energy column, one contains a potential energy column and one does not contain a energy column.

5.3.1 Gromacs: water particle

Free energy estimation of a water particle between to hydrophilic surfaces

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 17 for Coulomb, 20 for VDW

Length of Windows 10ns

System Size 3312 atoms

Temperature 300 K

Ensemble NVT

Volume 70.204 nm^3

Alchemical Pathway vacuum –> vdw –> vdw + coul

Missing Values None

Creator D. Wille

Donor Dominik Wille (harlor@web.de)

Date November 2018

License CC0 Public Domain Dedication

Similar free energy estimations can be found in [Schlaich2017].

`alchemtest.gmx.`**`load_water_particle_with_total_energy`**`()`
Load the Gromacs water particle with total energy dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

`alchemtest.gmx.`**`load_water_particle_with_potential_energy`**`()`
Load the Gromacs water particle with potential energy dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

`alchemtest.gmx.`**`load_water_particle_without_energy`**`()`
Load the Gromacs water particle without energy dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

5.4 Absolute Binding Free Energy of n-phenylglycinonitrile to T4 lysozyme

The dataset for computing the absolute binding free energy of n-phenylglycinonitrile to T4 lysozyme. The calculation has two legs: complex and ligand. In the complex leg, restraint is applied to the ligand and the coulombic as well as the Van der Waals interactions are decoupled sequentially. In the ligand leg, only the coulombic and Van der Waals interactions are decoupled.

5.4.1 Gromacs: n-phenylglycinonitrile in T4 lysozyme

Obtain the absolute binding free energy of the n-phenylglycinonitrile for T4 lysozyme by alchemically turning n-phenylglycinonitrile in T4 lysozyme and water into vacuum.

Notes

Data Set Characteristics:

Number of Legs 2 (Restraint, Coulomb, VDW for protein; Coulomb, VDW for water)

Number of Windows 11 for Restraint, 5 for Coulomb, 16 for VDW

Length of Windows 1ns for protein and 5ns for water

System Size 33005 atoms for protein and 2103 atoms for water

Temperature 300 K

Pressure 1 bar

Alchemical Pathway vdw + coul –> restraint + vdw + coul –> restraint + vdw –> restraint + vacuum

Reference Hydration Free Energy in protein -21.721 +- 0.089 kcal/mol

Reference Hydration Free Energy in water -7.679 +- 0.080 kcal/mol

Missing Values None

Energy unit kJ/mol

Time unit ps

Creator Z. Wu

Donor Zhiyi Wu (zhiyi.wu@bioch.ox.ac.uk)

Date March 2021

License CC0 Public Domain Dedication

This dataset was generated using tutorial , with the Gromacs molecular dynamics engine.

Data sourced from [Boyce2009].

`alchemtest.gmx.load_ABFE()`
Load the Gromacs ABFE dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

CHAPTER

SIX

AMBER DATASETS

Amber molecular dynamics simulation datasets.

The `alchemlyb.amber` module features datasets generated using the Amber molecular dynamics engine. They can be accessed using the following accessor functions:

load_bace_improper()	Load Amber Bace improper solvated vdw example
load_bace_example()	Load Amber Bace example perturbation.
load_simplesolvated()	Load the Amber solvated dataset.
load_invalidfiles()	Load the invalid files.
load_testfiles()	Load incomplete or wrongly formatted files to be used to test the AMBER parsers.

6.1 Amber: Small molecule thermodynamic integration free energy difference in water

Improper Bace solvated small molecule perturbation, alchemical vdw perturbation of ligand 1 into ligand 2. This example uses ligands CAT-13a to CAT-13m from [Wang2015b].

6.1.1 Notes

Data Set Characteristics:

Number of Legs 1 (vdw)

Number of Windows 12

Length of Windows 1ns

System Size 3920 atoms

Temperature 300 K

Pressure 1 bar

Alchemical Pathway vdw in ligand 1 –> vdw in ligand 2, softcore is used in vdw

Experimental Free Energy difference N/A

Missing Values None

Energy unit kcal/mol

Time unit ps

Date Jan 2018

Donor Silicon Therapeutics

License CC0 Public Domain Dedication

This dataset was generated using the Amber molecular dynamics engine.

`alchemtest.amber.load_bace_improper()`
Load Amber Bace improper solvated vdw example

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files for improper solvated vdw alchemical leg

Return type *Bunch*

6.2 Amber: Small molecule thermodynamic integration free energy difference in water

Bace complex and solvated small molecule perturbation, alchemical perturbation of ligand 1 into ligand 2. This example uses ligands CAT-13d to CAT-17a from [Wang2015a].

6.2.1 Notes

Data Set Characteristics:

Number of Legs 3 (decharge, vdw, recharge)

Number of Windows 5 for decharge, 12 for vdw, 5 for recharge

Length of Windows 1ns

System Size 46594 atoms (complex), 4115 atoms (solvated)

Temperature 300 K

Pressure 1 bar

Alchemical Pathway (decharge + vdw + recharge) in ligand 1 –> (decharge + vdw + recharge) in ligand 2, decharge, vdw, and recharge are running in parellel, soft core is used in vdw

Experimental Free Energy difference -0.26 kcal/mol

Missing Values None

Energy unit kcal/mol

Time unit ps

Date Jan 2018

Donor Silicon Therapeutics

License CC0 Public Domain Dedication

This dataset was generated using the Amber molecular dynamics engine.

`alchemtest.amber.load_bace_example()`
Load Amber Bace example perturbation.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by system and alchemical leg

Return type *Bunch*

6.3 Amber: Small molecule thermodynamic integration free energy difference in water

Small molecule perturbation in water, alchemically turned ligand 1 into ligand 2 in water. This example uses ligands 17124-1 to 18637-1 from [Wang2015c].

6.3.1 Notes

Data Set Characteristics:

Number of Legs 2 (charge, vdw)

Number of Windows 5 for charge, 12 for vdw

Length of Windows 1ns

System Size 5979 atoms

Temperature 300 K

Pressure 1 bar

Alchemical Pathway (charge + vdw) in ligand 1 –> (charge + vdw) in ligand 2, charge and vdw are running in parellel, soft core is used in vdw

Experimental Free Energy difference N/A

Missing Values None

Energy unit kcal/mol

Time unit ps

Date Oct 2017

Donor Silicon Therapeutics

License CC0 Public Domain Dedication

This dataset was generated using the Amber molecular dynamics engine.

`alchemtest.amber.load_simplesolvated()`
Load the Amber solvated dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

6.4 Amber TI invalid output files

Examples for file validation testing.

6.4.1 Notes

- no_useful_data.out.tar.bz2: file contains no useful data
- no_control_data.out.tar.bz2: file contains no control data
- no_temp0_setted.out.tar.bz2: file with Non-constant temperature
- no_free_energy_info.out.tar.bz2: file with no free energy section
- no_atomic_section.out.tar.bz2: file with no ATOMIC section
- no_results_section.out.tar.bz2: file with no RESULTS section

Deprecated since version 0.7: use *load_testfiles()* instead

alchemtest.amber.**load_invalidfiles**()

Load the invalid files.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the example of invalid data files
- 'DESCR': the full description of the dataset

Return type *Bunch*

Deprecated since version 0.7: use *load_testfiles()* instead

6.5 Amber: invalid/incomplete output files

Here we collected some invalid/incomplete AMBER output files that can be used to test specific part of the amber parser.

6.5.1 Notes

- no_atomic_section.out.bz2: AMBER output file without the ATOMIC section
- no_control_data.out.bz2: AMBER output file without the '2. CONTROL DATA FOR ' section
- no_dHdl_data_points.out.bz2: AMBER output file with TI active, but no DV/DL values,
- no_free_energy_info.out.bz2: AMBER output file without the settings regarding the free energy calculation
- no_results_section.out.bz2: AMBER output file without the RESULT section
- no_temp0_set.out.bz2: AMBER output file with temp0 not set
- no_useful_data.out.bz2: AMBER output file without useful data, truncated after the header
- none_in_mbar.out.bz2: AMBER output file with a wrongly formatted MBAR section. Specifically, a lambda value in a MBAR section has been altered, so it dowsn't match with the other MBAR sections and the expected lambda values (0.2500 –> 0.2550)

- not_finished_run.out.bz2: AMBER output file from an unterminated run
- high_number_of_mbar_windows.out.bz2: AMBER output file from a run with high number of MBAR lambdas
- no_spaces_around_equal.out.bz2: AMBER output file where there are no spaces around the '=' sign in the ' begin time read from' section
- no_starting_simulation_time.out.bz2: AMBER output file where the starting simulation time is not read

New in version 0.7.0.

alchemtest.amber.**load_testfiles**()
Load incomplete or wrongly formatted files to be used to test the AMBER parsers.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files
- 'DESCR': the full description of all the files

Return type *Bunch*

CHAPTER

SEVEN

NAMD DATASETS

NAMD molecular dynamics simulation datasets.

The `alchemlyb.namd` module features datasets generated using the NAMD molecular dynamics engine. They can be accessed using the following accessor functions:

load_tyr2ala()	Load the NAMD tyrosine to alanine mutation dataset.
load_idws()	Load the NAMD IDWS dataset.

7.1 NAMD: free energy of tyrosine to alanine mutation in aqueous solution

Free energy change from mutating a tyrosine (Y) residue into alanine (A) in the Ala-Tyr-Ala tripeptide in aqueous environment.

7.1.1 Notes

Data Set Characteristics:

Number of Legs 2 (forward Y–>A, backward A–>Y)

Number of Windows 20 for each leg

Length of Windows 1000 ps (each window interspersed with 200 ps equilibration)

System Size 1521 atoms

Temperature 300 K

Pressure 1 bar

Alchemical Pathway Point mutation of Tyr to Ala using dual topology hybrid molecule. Non-bonded interactions of perturbed atoms are scaled with their environment.

Experimental Free Energy difference N/A

Missing Values None

Energy unit kcal/mol

Time unit step

Date Oct 2017

Donor JC Gumbart

License CC0 Public Domain Dedication

This dataset was generated using the NAMD molecular dynamics engine.

alchemtest.namd.**load_tyr2ala**()
Load the NAMD tyrosine to alanine mutation dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

7.2 NAMD: free energy of dummy ethane to ethane "mutation" in aqueous solution

Free energy change from mutating an ethane molecule into an ethane molecule, turning a H atom into a methyl group and conversely. Expected free energy is zero, however the dataset is tiny (sufficient for testing purposes). Uses Interleaved Double-Wide Sampling (Hénin and Brannigan).

7.2.1 Notes

Data Set Characteristics:

Number of Legs 1 (forward mutation in water with IDWS sampling)

Number of Windows 11

Length of Windows 50 ps (each window interspersed with 5 ps equilibration)

System Size 1030 atoms

Temperature 300 K

Pressure 1 bar

Alchemical Pathway dummy mutation of ethane into ethane using dual topology hybrid molecule. Nonbonded interactions of perturbed atoms are scaled with their environment.

Theoretical Free Energy difference 0

Missing Values None

Energy unit kcal/mol

Time unit step

Date May 2021

Donor J Hénin

License CC0 Public Domain Dedication

This dataset was generated using the NAMD molecular dynamics engine.

alchemtest.namd.**load_idws**()
Load the NAMD IDWS dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

7.3 NAMD: free energy of tyrosine to alanine in vacuo

Free energy change from mutating a Tyr into Ala in vacuo. Uses Interleaved Double-Wide Sampling (Hénin and Brannigan). Each lambda window was run separately, and NAMD was interrupted and restarted multiple times, such that one window may span multiple fepout files.

Derived from NAMD FEP Tutorial, available at: https://www.ks.uiuc.edu/Training/Tutorials/namd/FEP/

7.3.1 Notes

Data Set Characteristics:

Number of Legs 1 (forward mutation with IDWS sampling)

Number of Windows 11

Length of Windows 50 ps

System Size 57 atoms

Temperature 300 K

Alchemical Pathway Mutation of Tyr into Ala using hybrid molecule. Nonbonded interactions of perturbed atoms are scaled with their environment.

Missing Values None

Energy unit kcal/mol

Time unit step

Date August 2021

Donor Thomas T. Joseph

License CC0 Public Domain Dedication

This dataset was generated using the NAMD molecular dynamics engine.

`alchemtest.namd.load_restarted()`
Load the NAMD IDWS dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

7.4 NAMD: free energy of tyrosine to alanine in vacuo

Free energy change from mutating a Tyr into Ala in vacuo. Uses Interleaved Double-Wide Sampling (Hénin and Brannigan). Each lambda window was run separately, and NAMD was interrupted and restarted multiple times, such that one window may span multiple fepout files.

Derived from NAMD FEP Tutorial, available at: https://www.ks.uiuc.edu/Training/Tutorials/namd/FEP/

This calculation was run from lambda = 1.0 to lambda = 0.0, because it is possible for an IDWS calculation in NAMD to be run this way.

7.4.1 Notes

Data Set Characteristics:

Number of Legs 1 (forward mutation with IDWS sampling)

Number of Windows 11

Length of Windows 50 ps

System Size 57 atoms

Temperature 300 K

Alchemical Pathway Mutation of Tyr into Ala using hybrid molecule. Nonbonded interactions of perturbed atoms are scaled with their environment.

Missing Values None

Energy unit kcal/mol

Time unit step

Date September 2021

Donor Thomas T. Joseph

License CC0 Public Domain Dedication

This dataset was generated using the NAMD molecular dynamics engine.

`alchemtest.namd.load_restarted_reversed()`

Load the NAMD IDWS dataset, run from lambda = 1 -> 0, with interruptions and restarts.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

CHAPTER

EIGHT

GOMC DATASETS

GOMC Monte Carlo simulation datasets.

The `alchemlyb.gomc` module features datasets generated using the GPU Optimized Monte Carlo (GOMC) simulation engine. They can be accessed using the following accessor functions:

load_benzene()	Load the GOMC benzene dataset.

8.1 Simple TI and FEP

The data sets contain derivatives of the Hamiltonian (TI) and free energy perturbation (FEP) data suitable for processing with FEP estimators as well as BAR/MBAR. Individual λ windows were run independently.

8.1.1 GOMC: Benzene in water

Hydration free energy of benzene using the *TraPPE-EH* [Raj2007] model and the SPC water model.

Notes

Data Set Characteristics:

Number of Legs 2 (Coulomb, VDW)

Number of Windows 7 for Coulomb, 15 for VDW

Length of Windows 50 million Monte Carlo steps

System Size 1001 molecules

Temperature 298 K

Pressure 1 bar

Alchemical Pathway vacuum –> vdw –> vdw + coul

Experimental Hydration Free Energy -0.90 $\pm$ 0.2 kcal/mol

Missing Values None

Energy unit kJ/mol

Time unit Monte Carlo steps

Creator M. Soroush Barhaghi

Donor Mohammad Soroush Barhaghi (m.soroush@wayne.edu)

Date July 2019

License CC0 Public Domain Dedication

This dataset was generated using GOMC Monte Carlo simulation engine.

Experimental value sourced from [Mobley2013b].

alchemtest.gomc.**load_benzene**()

Load the GOMC benzene dataset.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files by alchemical leg
- 'DESCR': the full description of the dataset

Return type *Bunch*

CHAPTER

NINE

GENERIC DATASETS

Simulation datasets in any form.

The `alchemlyb.generic` module features datasets that are MD engine free and can adopt any form. They can be accessed using the following accessor functions:

load_MBAR_BGFS()	Load data set that will fail the MBAR adaptive solver but could done by BGFS.

9.1 Difficult case for the adaptive MBAR solver

The `pymbar.mbar.MBAR` can have difficulty in solving some dataset with the *adaptive* method, where *BFGS* is needed.

The usage is like this

```
>>> import numpy as np
>>> from pymbar import MBAR
>>> from alchemtest.generic import load_MBAR_BGFS
>>> u_nk = np.load(load_MBAR_BGFS()['data']['u_nk'])
>>> N_k = np.load(load_MBAR_BGFS()['data']['N_k'])
>>> solver_options = {"maximum_iterations":10000,"verbose":True}
>>> solver_protocol = {"method":"adaptive","options":solver_options}
>>> mbar = MBAR(u_nk, N_k, solver_protocol=(solver_protocol,))
>>> results, errors = mbar.getFreeEnergyDifferences()
```

Which will give the `pymbar.utils.ParameterError`

```
>>> solver_options = {"maximum_iterations":10000,"verbose":True}
>>> solver_protocol = {"method":"BFGS","options":solver_options}
>>> mbar = MBAR(u_nk, N_k, solver_protocol=(solver_protocol,))
>>> results, errors = mbar.getFreeEnergyDifferences()
```

Which will work.

9.1.1 generic: MBAR solver stability test

u_nk and N_k files that could be calculated with MBAR using BGFS method but not the adaptive method.

Notes

Data Set Characteristics:

Number of Legs N/A

Number of Windows 1

Length of Windows N/A

System Size N/A

Temperature N/A

Pressure 1 N/A

Alchemical Pathway N/A

Missing Values None

Energy unit N/A

Time unit N/A

Creator Z. Wu

Donor Ryan S. DeFeverU (defever@nd.edu)

Date Oct 2021

License CC0 Public Domain Dedication

This dataset was provided by @rsdefever on Github , downloaded and uploaded by Zhiyi Wu (@xiki-tempula).

`alchemtest.generic.load_MBAR_BGFS()`

Load data set that will fail the MBAR adaptive solver but could done by BGFS.

Returns

data – Dictionary-like object, the interesting attributes are:

- 'data' : the data files for u_nk and N_k

Return type *Bunch*

BIBLIOGRAPHY

[Mobley2013] Mobley, David L. (2013). Experimental and Calculated Small Molecule Hydration Free Energies. UC Irvine: Department of Pharmaceutical Sciences, UCI. Retrieved from: http://escholarship.org/uc/item/6sd403pz

[Muddana2014a] H. Muddana, A. Fenley, D. Mobley, and M. Gilson. The SAMPL4 host–guest blind prediction challenge: an overview. Journal of Computer-Aided Molecular Design, 28(4):305–317, 2014. PMID: 24599514. DOI: 10.1007/s10822-014-9735-1.

[Monroe2014a] J. Monroe and M. Shirts. Converging free energies of binding in cucurbit[7]uril and octa-acid host-guest systems from SAMPL4 using expanded ensemble simulations. Journal of Computer-Aided Molecular Design, 28(4):401–415, 2014. PMID: 24610238 DOI: 10.1007/s10822-014-9716-4.

[Muddana2014b] H. Muddana, A. Fenley, D. Mobley, and M. Gilson. The SAMPL4 host–guest blind prediction challenge: an overview. Journal of Computer-Aided Molecular Design, 28(4):305–317, 2014. PMID: 24599514. DOI: 10.1007/s10822-014-9735-1.

[Monroe2014b] J. Monroe and M. Shirts. Converging free energies of binding in cucurbit[7]uril and octa-acid host-guest systems from SAMPL4 using expanded ensemble simulations. Journal of Computer-Aided Molecular Design, 28(4):401–415, 2014. PMID: 24610238 DOI: 10.1007/s10822-014-9716-4.

[Muddana2014c] H. Muddana, A. Fenley, D. Mobley, and M. Gilson. The SAMPL4 host–guest blind prediction challenge: an overview. Journal of Computer-Aided Molecular Design, 28(4):305–317, 2014. PMID: 24599514. DOI: 10.1007/s10822-014-9735-1.

[Schlaich2017] Alexander Schlaich, Julian Kappler, and Roland R. Netz. Hydration Friction in Nanoconfinement: From Bulk via Interfacial to Dry Friction. Nano Lett., 2017, 17 (10), pp 5969–5976. DOI: 10.1021/acs.nanolett.7b02000.

[Boyce2009] Boyce, S.E., Mobley, D.L., Rocklin, G.J., Graves, A.P., Dill, K.A., Shoichet, B.K. (2009) Predicting Ligand Binding Affinity with Alchemical Free Energy Methods in a Polar Model Binding Site. J. Mol. Biol. 394, 747–7636

[Wang2015b] L. Wang, Y. Wu, Y. Deng, B. Kim, L. Pierce, G. Krilov, D. Lupyan, S. Robinson, M. K. Dahlgren, J. Greenwood, D. L. Romero, C. Masse, J. L. Knight, T. Steinbrecher, T. Beuming, W. Damm, E. Harder, W. Sherman, M. Brewer, R. Wester, M. Murcko, L. Frye, R. Farid, T. Lin, D. L. Mobley, W. L. Jorgensen, B. J. Berne, R. A. Friesner, and R. Abel. Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. Journal of the American Chemical Society, 137(7):2695–2703, 2015. PMID: 25625324. DOI: 10.1021/ja512751q.

[Wang2015a] L. Wang, Y. Wu, Y. Deng, B. Kim, L. Pierce, G. Krilov, D. Lupyan, S. Robinson, M. K. Dahlgren, J. Greenwood, D. L. Romero, C. Masse, J. L. Knight, T. Steinbrecher, T. Beuming, W. Damm, E. Harder, W. Sherman, M. Brewer, R. Wester, M. Murcko, L. Frye, R. Farid, T. Lin, D. L. Mobley, W. L. Jorgensen, B. J. Berne, R. A. Friesner, and R. Abel. Accurate and reliable prediction of relative ligand

binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. Journal of the American Chemical Society, 137(7):2695–2703, 2015. PMID: 25625324. DOI: 10.1021/ja512751q.

[Wang2015c] L. Wang, Y. Wu, Y. Deng, B. Kim, L. Pierce, G. Krilov, D. Lupyan, S. Robinson, M. K. Dahlgren, J. Greenwood, D. L. Romero, C. Masse, J. L. Knight, T. Steinbrecher, T. Beuming, W. Damm, E. Harder, W. Sherman, M. Brewer, R. Wester, M. Murcko, L. Frye, R. Farid, T. Lin, D. L. Mobley, W. L. Jorgensen, B. J. Berne, R. A. Friesner, and R. Abel. Accurate and reliable prediction of relative ligand binding potency in prospective drug discovery by way of a modern free-energy calculation protocol and force field. Journal of the American Chemical Society, 137(7):2695–2703, 2015. PMID: 25625324. DOI: 10.1021/ja512751q.

[Mobley2013b] Mobley, David L. (2013). Experimental and Calculated Small Molecule Hydration Free Energies. UC Irvine: Department of Pharmaceutical Sciences, UCI. Retrieved from: https://escholarship.org/uc/item/6sd403pz

[Raj2007] Neeraj Rai and J. Ilja Siepmann (2007). The Journal of Physical Chemistry B, 111 (36), 10790-10799 DOI: 10.1021/jp073586l

PYTHON MODULE INDEX

a

INDEX