NMRlipids Databank

Anne Kiirikki[1,*] **and O. H. Samuli Ollila**[1,*]

[1]University of Helsinki, Institute of Biotechonology, Helsinki, Finland
[*]samuli.ollila@helsinki.fi

ABSTRACT

We present a databank of lipid bilayer simulations from the NMRlipids open collaboration project.

Introduction

The demand for sharing and reusing of MD simulation data is increasing, but practical solution remains unclear due to unsolved issues in data storage and indexing. Here we present a solution for lipid bilayers based on overlay databank structure.

Results

Quality evaluation of force fields

The quality of each simulation is evaluated against NMR order parameters. Indexing of experimental data to automatize evaluation is currently in progress by Anne Kiirikki.

Water permeability across membranes

Averaging water density profiles over all simulations in the databank enables us to calculate the barrier for the water penetration through lipid bilayers.

Spin relaxation rates of confined water close to membranes

Usefulness of the databank beyond MD simulation experts is demonstrated by analysing water spin relaxation times close to bilayers which are used in MRI imaging.

Discussion

Methods

Structure of the databank

NMRlipids databank is a overlay databank composed of files that contain all the essential information on the molecular dynamics simulation trajectories that are listed in the databank. The location publicly avaiable raw MD simulation data is flexible. The NMRlipids databank files contain all the essential information on the original data that enable wide range of applications.

Each simulation system is identified using the hash of original trajectory and topology file. The index files are stored in folder structure based on the identity codes of the simulations.

Analysis of the data in databank

Index files in the datanbank contain all the essential information to perform analyses from the simulations. Systems under interest can be selected automatically browsing the index files and filtering the desired properties.

Analysis results can be most conveniently stored to a new datanbank with identical indexing as the original databank. The results can be then easily shared and browsed indentically to the original databank.

Indexing the simulation data

AddData.py is a script that builds a database that contains a dictionary file and analysis data of each simulation. The dictionary file contains information about the simulation. The script also calculates order parameters of all CH bonds of the lipids in the simulation. To add a simulation it must be first uploaded to Zenodo (www.zenodo.org). The trajectory and topology files of the simulation are downloaded to the working directory from Zenodo but these are not saved into the database. To add a simulation to the database the user has to give some essential information about the simulation. This is done by writing a info

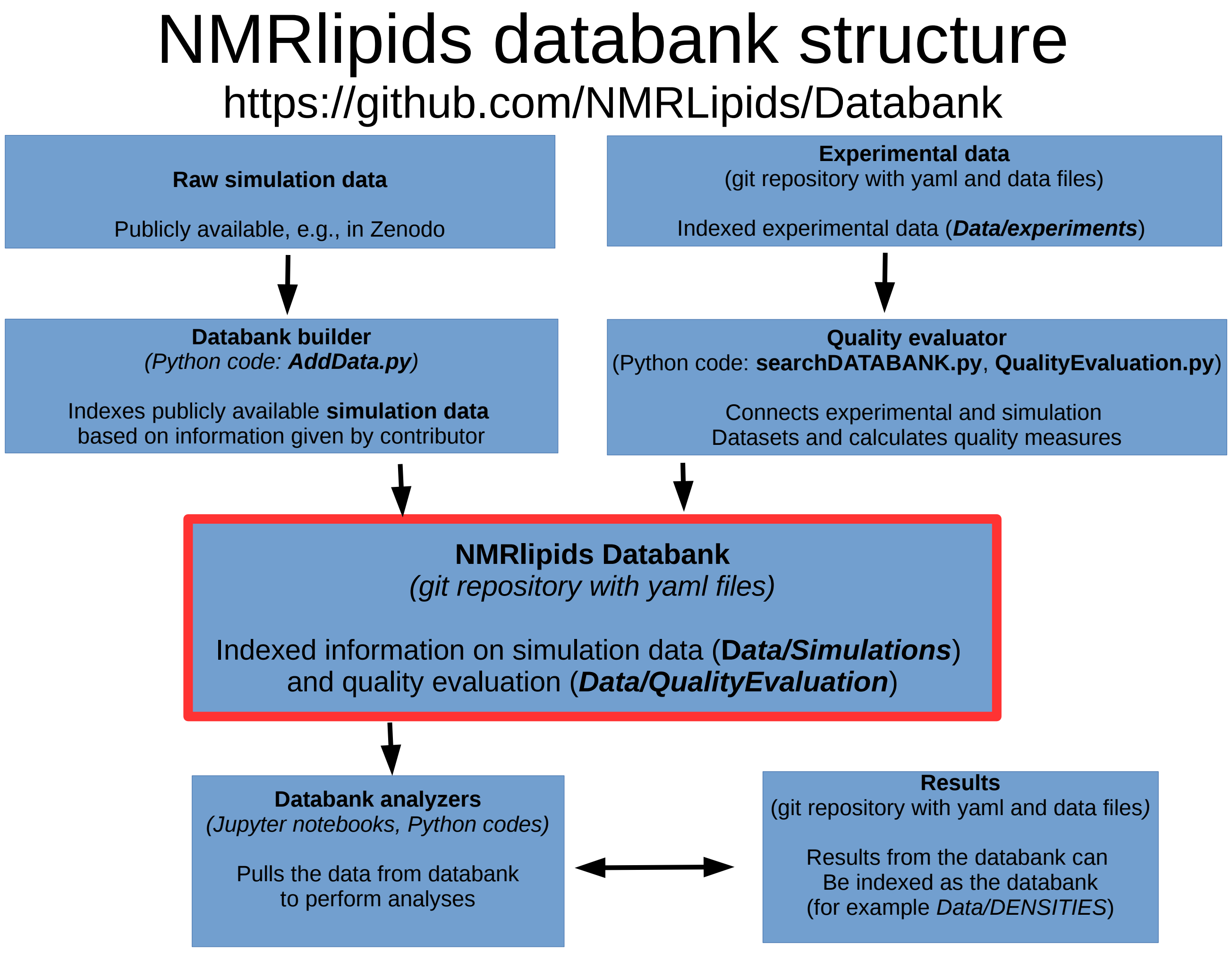
NMRlipids databank structure
https://github.com/NMRLipids/Databank
Raw simulation data
Publicly available, e.g., in Zenodo
Experimental data
(git repository with yaml and data files)
Indexed experimental data (***Data/experiments***)
Databank builder
*(Python code: **AddData.py**)*
Indexes publicly available **simulation data** based on information given by contributor
Quality evaluator
(Python code: **searchDATABANK.py**, **QualityEvaluation.py**)
Connects experimental and simulation Datasets and calculates quality measures
NMRlipids Databank
(git repository with yaml files)
Indexed information on simulation data (***Data/Simulations***) and quality evaluation (***Data/QualityEvaluation***)
Databank analyzers
(Jupyter notebooks, Python codes)
Pulls the data from databank to perform analyses
Results
(git repository with yaml and data files)
Results from the databank can Be indexed as the databank (for example *Data/DENSITIES*)

Figure 1. Structure of the NMRlipids databank

file (*INFO.yaml) which is passed to AddData.py.

AddData.py requires GROMACS and MDAnalysis library to be installed.

Run AddData.py as follows:

```
python3 AddData.py -f exampleINFO.yaml
```

There is also a script runAddData.sh that can be used to loop over several info files to add many simulations at one go.

A simulation dictionary contains information about the simulation. Some of the information is provided by the user. The numbers of lipid molecules, solvent and ions are automatically read from the files and so are the simulation temperature and trajectory length. All this information is saved to a file named README.yaml.

User input

Dictionary variables defined by the user are listed here. The necessary parameters required for the analyses are described first and are marked as ”compulsory”. Then the parameters used to describe the data for further usage and analyses are described. While only compulsory parameters are required for the databank entry, strong recommendation is that all the possible parameters would be given to maximize the possibilities to upcycle the contributed data.

DOI (compulsory)

Give DOI identity from where the simulation files are located. Current databank works only for the data in Zenodo, but other potential sources are may be implemented in the future. Note that the DOI must point to a specific version of dataset in Zenodo, i.e., DOI pointing to all versions of certain dataset does not work.

SOFTWARE (compulsory)

Give the name of software used to run the simulation. The options are GROMACS, AMBER, NAMD, CHARMM and OPENMM. So far, only simulations run with GROMACS are accepted by the script.

TRJ (compulsory)

Give the name of the trajectory file that is found from the DOI given above.

TPR (compulsory)

Give the name of the file with topology information (tpr file in the case of Gromacs) that is found from the DOI given above.

PREEQTIME (compulsory)

Give the time simulated before the uploaded trajectory in nanoseconds. For example, if you upload 100-200 ns part of total 200 ns simulation, this should value should be 100.

TIMELEFTOUT (compulsory)

Give the time that should be considered as an equilibration period in the uploaded trajectory. Frames before the give time will be discarded in the analysis. For example, if you upload 0-200 ns part of total 200 ns simulation where the first 100 ns should be considered as an equilibration, this value should be 100.

Molecule names (compulsory)

In the databank, each molecule has a unique abbreviation (see table 1). You need to give the residue names of these molecules corresponding your simulation. If atoms in a lipid belong to different residues (typical situation in Amber force fields), give the name of the head group residue here, and add the residue name of each atom to the third column in the mapping file (see below). If your simulation contains molecules that are not yet in the databank, you need to define the abbreviation and add molecules to the lipids_dict, molecules_dict, molecule_numbers_dict and molecule_ff_dict in the AddData.py script, as well as to table 1.

MAPPING_DICT (compulsory)

For the analysis, we need to know the names of atoms in your system. These are defined using the mapping file convention introduced in `http://nmrlipids.blogspot.com/2015/03/mapping-scheme-for-lipid-atom-names-for.html`, where first column gives the universal atom name, second gives the atom name in force field, and third gives the residue name if not the same for all atoms. The name of the mapping file for each molecule needs to be given in dictionary format. The dictionary the key is the molecule name (abbreviation listed in table 1) and the value is the name of the mapping file. The already existing mapping files can be found from the directory named "mapping_files". If a mapping file for your molecule(s) do not exist, you need to construct one and add to the "mapping_files" directory.

Abbreviation	Molecule name
POPC	1-palmitoyl-2-oleoyl-sn-glycero-3-phosphocholine
POPG	1-palmitoyl-2-oleoyl-sn-glycero-3-phosphoglycerol
POPS	1-palmitoyl-2-oleoyl-sn-glycero-3-phospho-L-serine
POPE	1-palmitoyl-2-oleoyl-sn-glycero-3-phosphoethanolamine
CHOL	cholesterol
DHMDMAB	dihexadecyldimethylammonium
POT	potassium ion
SOD	sodium ion
CLA	chloride ion
CAL	calcium ion
SOL	water

Table 1. Abbreviations used in the databank

DIR_WRK (compulsory)
Give the path of the working directory in your local computer. The trajectory and topology files will be downloaded to this trajectory, and temporary files created during processing will be stored here.

UNITEDATOM_DICT (compulsory for united atom trajectories)
Order parameters from united atom simulations are calculated using buildH code (`https://github.com/patrickfuchs/buildH`). For united atom simulations, you need to tell how hydrogens are added based on definitions in the dic_lipids.py dictionary. This is done by giving a dictionary where the key is the molecule name (abbreviation listed in table 1) and the value is the correct dictionary key in dic_lipids.py. If correct dictionary key is not yet found, you need to add to dic_lipids.py. In the case of an all atom simulation, UNITEDATOM is left empty.

PUBLICATION
Give reference to a publication(s) related to the data.

AUTHORS_CONTACT
Give the name and email of the main author(s) of the data.

SYSTEM
Give description of system in free format. For example ”POPC with cholesterol at 301K”.

SOFTWARE_VERSION
Give the version of the software used.

FF
Give the name of the force field used used in the simulation.

FF_SOURCE
Describe the source of the force field parameters. For example, CHARMM-GUI, link to webpage where parameters were downloaded, or citation to a paper.

FF_DATE
Give the date when parameters were accessed or created. The format is day/month/year.

Individual force field names for molecules
In some cases special force fields are used for certain molecules. For example, non-standard parameters for ions or other molecules have been used. These can be specified giving forcefield names separately for individual molecules. These can be given as parameters named as FF+[abbreviation from table 1], i.e., FFPOPC, FFPOT, FFSOL etc.

CPT (Gromacs)
Give the name of the Gromacs checkpoint file that is found from the DOI given above. CPT stands for the name of the cpt file.

LOG (Gromacs)
Give the name of the Gromacs log file that is found from the DOI given above.

TOP (Gromacs)
Give the name of the Gromacs top file that is found from the DOI given above.

Automatically analyzed parameters
The following parameters are read automatically from the trajectory and topology files.

Molecule numbers
Numbers of lipid molecules (NPOPC, NPOPG, etc.) per membrane leaflet are calculated by determining on which side of the center of mass of the membrane the center of mass of the head group of each lipid molecule is located.

Numbers of other molecules such as solvent and ions (NSOL, NPOT, NSOD, etc.) are read from the topology file.

Temperature
Temperature of the simulation is read from the topology file.

Trajectory length
The length of a trajectory is read from the trajectory file.

Date of running
The date of running the script is saved to the README.yaml file.

Acknowledgements

Author contributions statement

Must include all authors, identified by initials, for example: A.A. conceived the experiment(s), A.A. and B.A. conducted the experiment(s), C.A. and D.A. analysed the results. All authors reviewed the manuscript.

Additional information