ContinuumElectrostatics

User Manual
Last update: June 17, 2014

1 Introduction

Proteins contain residues, cofactors and ligands that bind or release protons depending on the current pH and the interactions with their molecular environment. These titratable residues, cofactors and ligands will be referred to as sites. The titration of proteins is often difficult to study experimentally because the available methods, such as calorimetry, cannot determine protonation states of individual sites. The knowledge of these individual protonation states is crucial for understanding of many important processes, for example enzyme catalysis.

The ContinuumElectrostatics module extends the functionality of the pDynamo library with a Poisson-Boltzmann continuum electrostatic model that allows for calculations of protonation states of individual sites. The module provides an interface between pDynamo and the external solver of the Poisson-Boltzmann equation, MEAD. MEAD is a program developed by Donald Bashford and extended by Timm Essigke and Thomas Ullmann. The electrostatic energy terms obtained with MEAD can be used to calculate energies of all possible protonation states of the protein of interest. However, analytic evaluation of protonation state energies is only possible for proteins with only a few titratable sites. The ContinuumElectrostatics module also provides an interface to the GMCT program by Matthias Ullmann and Thomas Ullmann that can be used to sample protonation state energies using a Monte Carlo method. The GMCT interface allows for studying the titration of larger proteins.

In the present document, I focus on the practical side of the electrostatic calculations. Other people have done a great work to develop the theory and computational methods the ContinuumElectrostatics module is based on.

2 Copying

The module is distributed under the CeCILL Free Software License, which is a French equivalent of the GNU General Public License. For details, see the files `Licence_CeCILL_V2-en.txt` (or `Licence_CeCILL_V2-fr.txt` for the French version).

3 Goals of the ContinuumElectrostatics module

I wrote the ContinuumElectrostatics module primarily as a pretext to learn how the Poisson-Boltzmann model works in detail. I used this model very often during my studies on enzyme catalysis but never had time nor will to learn the details of the theory that was behind. I also wanted to better explore the pDynamo library, understand its programming concepts and finally extend it with something useful. Last but not least, I saw some room for improvement in the previously used tools and scripts.

The ContinuumElectrostatics module is similar in behaviour to the `multiflex2qmpb.pl` program, which is part of the QMPB package written by Timm Essigke. The approach taken here is most compatibile with the original approach by Donald Bashford. The key difference is that the treatment of multiprotic sites, such as histidine, is improved.

4 Installation and configuration

Before the installation of the ContinuumElectrostatics module, it is necessary to have:

- pDynamo 1.8.0
- Python 2.7
- GCC (any version should be fine)
- Extended MEAD 2.3.0
- GMCT 1.2.3

Extended MEAD and GMCT can be found on the website of Thomas Ullmann:
`http://www.bisb.uni-bayreuth.de/People/ullmannt/index.php?name=software`
Download the two packages and follow their respective installation instructions. The ContinuumElectrostatics module requires for its functioning two programs from the MEAD package, namely `my_2diel_solver` and `my_3diel_solver`, and the GMCT's main program, `gmct`.

In the next step, check out the latest source code of the module. Note that for checking out the source code you should have Subversion installed as well.

```
svn checkout
    http://pdynamo-extensions.googlecode.com/svn/trunk/ContinuumElectrostatics/
```

Some parts of the module implementing the state vector are written in C and therefore have to be compiled before use. In the future, I plan to shift some more parts of the code from Python to C.

Go to the subdirectory `extensions/csource` and edit the Makefile. Only the uppermost line has to be changed. It defines the directory where you have installed pDynamo. After editing, close the file and type in `make` to compile the C object file.

Go to the subdirectory `extensions/pyrex` and again edit the Makefile. Change the line starting from ”INC2” to the location of your pDynamo installation. Close the file and type in `make`. It should generate a dynamically linked library `StateVector.so` in the `ContinuumElectrostatics` subdirectory.

At this point, the installation is complete.

Before using the module, you have to set the environment variable
`PDYNAMO_CONTINUUMELECTROSTATICS` to the directory where you have checked out the source code. Also, you have to add this directory to the `PYTHONPATH` variable. For example, you can do it this way (in Bash):

```
export
    PDYNAMO_CONTINUUMELECTROSTATICS=/home/mikolaj/devel/ContinuumElectrostatics

export PYTHONPATH=$PYTHONPATH:$PDYNAMO_CONTINUUMELECTROSTATICS
```

5 Usage

After the installation, it may be worth looking at some of the test cases. I will explain the functioning of the module based on the test case ”sites2”. This test uses a trivial polypeptide with only two titratable sites, histidine and glutamate. The test ”histidine” uses only one site. The other tests use real-life, although small proteins.

5.1 Setup of the protein model

The electrostatic model used by the ContinuumElectrostatics module requires that the protein of interest is described by the CHARMM energy model. In the first step, prepare CHARMM topology (psf) and coordinate (crd) files. The preperation can be done using the CHARMM program or VMD. During the preparation of the protein model, all titratable residues in the protein should be set to their standard protonation states at pH = 7, i.e. aspartates and glutamates deprotonated, histidines doubly protonated, other residues protonated. The topology, coordinate and parameter files are loaded at the beginning of the script `sites2.py`:

```
par_tab = ["charmm/toppar/par_all27_prot_na.inp", ]

mol  = CHARMMPSFFile_ToSystem ("charmm/testpeptide_xplor.psf", isXPLOR = True,
    parameters = CHARMMParameterFiles_ToParameters (par_tab))

mol.coordinates3 = CHARMMCRDFile_ToCoordinates3 ("charmm/testpeptide.crd")
```

5.2 Setup of the continuum electrostatic model

In the second step, a continuum electrostatic model is created:

```
ce_model = MEADModel (meadPath = "/home/mikolaj/local/bin/", gmctPath =
    "/home/mikolaj/local/bin/", scratch = "scratch", nthreads = 2)
```

The parameter ”meadPath” tells the directory where the MEAD programs, my_2diel_solver and my_3diel_solver, are located. The parameter ”gmctPath” is the location of the gmct program. If none of these directories are given, /usr/bin is assumed by default. The parameter ”scratch” tells the directory where the MEAD job files and output files will be written to. If not present, this directory will be created. The last parameter, ”nthreads”, defines the number of threads to be used. By default nthreads=1, which means serial run. Note that ”nthreads” can be any natural number and that the calculations scale linearly with the number of threads.

In the next step, the continuum electrostatic model is initialized:

```
ce_model.Initialize (mol)
```

The initialization means partitioning of the protein into titratable sites and a non-titratable background. It also means generating model compounds. At this point, however, the input files for MEAD are not written and only the necessary data structures inside the MEADModel object are created. The Initialize method takes at least one argument, which indicates the CHARMM-based protein model.

The next two lines print out summaries of the continuum electrostatic model and the titratable sites:

```
ce_model.Summary ()
ce_model.SummarySites ()
```

After the model has been initialized, the input files necessary for calculations in MEAD can be written to the scratch directory:

```
ce_model.WriteJobFiles (mol)
```

6 References

MEAD website:

`http://stjuderesearch.org/site/lab/bashford/`

Extended MEAD website:

`http://www.bisb.uni-bayreuth.de/People/ullmannt/index.php?name=extended-mead`

Doctoral thesis of Timm Essigke:

`https://epub.uni-bayreuth.de/655/`

7 Test cases

8 To-do list

- Move parts of WriteJobFiles to the instance class
- Rename variables containing filenames so that they start from ”file”
- Coordinates from the FPT file should have their own data structure
- Efficiency improvements during writing job files
- Use arrays instead of lists for interactions (Real1DArray or SymmetricMatrix)
- Have a column of ETA (Estimated Time for Accomplishment) in MEAD calculations
- Optionally convert kcal/mol (MEAD units) to kJ/mol (pDynamo units)
- Make use of the pqr2SolvAccVol program to speed up the calculations a little bit
- The function calculating Gmicro should be written in C