Pcetk

A pDynamo-Based Continuum Electrostatic Toolkit

Mikolaj Feliks

mikolaj.feliks@gmail.com

Institut de Biologie Structurale, Grenoble

Last updated: September 23, 2015

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[11]. These titratable residues, cofactors and ligands will be referred to as sites. Each site exists in at least two charge forms, usually the protonated form and deprotonated form. Different charge forms of a site are called instances. The titration of proteins is often difficult to study experimentally. Moreover, 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[4].

The *Pcetk* toolkit extends the pDynamo library by Martin Field[7] with a Poisson-Boltzmann continuum electrostatic model that allows for the calculation of protonation states in proteins[2;9;11]. The toolkit connects pDynamo to the external solver of the Poisson-Boltzmann equation, MEAD[1]. The idea to employ MEAD for the calculation of electrostatic energy terms is similar to how pDynamo interfaces with ORCA to calculate quantum chemical energies and properties.

After the calculation of electrostatic energy terms by MEAD, there are three ways to

calculate the probabilities of protonation states. First, they can be calculated by using the statistical mechanical partition function, which is, however, only possible for small proteins. Second, the probabilities can be estimated by using the in-house version of the Metropolis Monte Carlo method[3]. Third, they can be estimated by the external Monte Carlo sampling program, GMCT[12;13]. The letter two methods make it possible to study the titration of larger proteins.

2 Copying

The toolkit is distributed under the CeCILL Free Software License, which is a French equivalent of the GNU General Public License. More information can be found in files `Licence_CeCILL_V2-en.txt` or `Licence_CeCILL_V2-fr.txt` (the French version).

3 Goals of the *Pcetk* toolkit

I wrote *Pcetk* primarily as a pretext to learn how the Poisson-Boltzmann model and Monte Carlo sampling work in detail. I used this model very often during my PhD studies on enzyme catalysis but never had time nor pressure to learn the details of the theory that was behind. I also wanted to better explore pDynamo, understand its programming concepts and finally extend it with something useful. Finally, I saw some room for improvement in the previously used tools and scripts.

The *Pcetk* toolkit aims at combining the functionalities of earlier packages for studying the titration of proteins, such as MEAD, QMPB and GMCT. MEAD is a solver of the Poisson-Boltzmann equation, written originally by Donald Bashford and extended by Timm Essigke[6] and Thomas Ullmann[13]. QMPB is a collection of Perl scripts for the preparation of continuum electrostatic models of proteins, written by Timm Essigke[6]. GMCT is a Monte Carlo sampling program, written by Matthias Ullmann and extended

by Thomas Ullmann[12]. In *Pcetk*, all these functionalities can be accessed by the user in a convenient, Python-based scripting environment. The "engine" of *Pcetk* that handles the actual calculation is located in the underlying layer programmed in C. The two layers are glued together with Cython. Therefore, the trade-off between the convenience of use of Python and the speed of compiled languages is well balanced in the toolkit.

4 Installation and configuration

Before the installation of *Pcetk*, it is necessary to have:

- pDynamo 1.8.0
- Python 2.7 (including header files; python2.7-dev package in Debian)
- PyYAML 3.10 (python-yaml package in Debian)
- Cython 0.15.1 (optionally; for recompiling Cython sources)
- GCC (any version should be fine)
- Extended-MEAD 2.3.0
- GMCT 1.2.3 (optionally)

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 toolkit requires for its functioning two programs from the Extended-MEAD package. These are `my_2diel_solver` and `my_3diel_solver`. In order to use GMCT for Monte Carlo sampling, it is necessary to only have the GMCT's main program, `gmct`.

In the next step, download the latest source code of the toolkit from:
`http://github.com/mfx9/pcetk/archive/master.zip`
Alternatively, clone the repository from GitHub:

```
git clone http://github.com/mfx9/pcetk.git
```

Some parts of the toolkit's code are written in C or Cython and therefore have to be compiled before use. These parts include three submodules, namely `StateVector`, `EnergyModel` and `MCModelDefault`.

Go to directory `extensions/cython/`. Edit the first line of `Makefile` starting from "PDYNAMO_PCORE". The line should define the location of the pCore module of the present pDynamo installation. After editing the file, run "make". If you also want to recompile the Cython sources, precede "make" by running "make clean_all". If the Cython compilation fails, it may be necessary to manually specify the location of the pCore module in the `cython_compile.py` script. Finally, run "make install" to install the compiled library.

At this point, the installation is complete.

Before using the toolkit, the environment variable `PDYNAMO_PCETK` should be assigned to point to the root directory of the toolkit. This directory should also be added to the `PYTHONPATH` variable. This can be done in the following way (in Bash):

```
export PDYNAMO_PCETK=/home/mikolaj/devel/pcetk
export PYTHONPATH=$PYTHONPATH:$PDYNAMO_PCETK
```

5 Usage

After finishing the installation, it may be worth looking at some of the test cases. The functioning of the toolkit will be explained based on the test case "twosites". This test uses a trivial polypeptide with only two titratable sites, histidine and glutamate. The other tests use real-life, although small proteins.

By default, the scratch directory is not emptied after completing the run. Starting the job for the second time will cause reading of the existing scratch files instead of running the calculations anew. To avoid this behavior, remove the scratch directory or declare another one during the setup of the continuum electrostatic model.

Notice! The energies calculated by *Pcetk* are given in $^{\mathrm{kcal}}/_{\mathrm{mol}}$, in contrast to $^{\mathrm{kJ}}/_{\mathrm{mol}}$, which are normally used in pDynamo.

5.1 Setup of the protein model

The electrostatic model used by the toolkit requires that the protein of interest is described by the CHARMM energy model[10]. In the first step, prepare CHARMM topology (PSF) and coordinate (CRD) files. The preparation can be done using, for example, CHARMM[5] or VMD[8]. 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 should be deprotonated, histidines doubly protonated and other residues protonated. Topology, coordinate and parameter files are loaded at the beginning of script `twosites.py`:

```
parameters = ["charmm/toppar/par_all27_prot_na.inp", ]
mol = CHARMMPSFFile_ToSystem ("charmm/testpeptide_xplor.psf",
   isXPLOR=True, parameters=CHARMMParameterFiles_ToParameters
   (parameters))
mol.coordinates3 = CHARMMCRDFile_ToCoordinates3
   ("charmm/testpeptide.crd")
```

5.2 Setup of the continuum electrostatic model

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

```
cem = MEADModel (system=mol, pathMEAD="/home/mikolaj/local/bin/",
   pathScratch="mead", nthreads=2)
```

The first parameter, `system`, indicates the CHARMM-based protein model. Parameter `pathMEAD` specifies the directory where the MEAD programs, `my_2diel_solver` and `my_3diel_solver`, are located. If this directory is not given, "/usr/local/bin" is assumed by default. Parameter `pathScratch` defines 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 use during parallel calculations. 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. Parallelization is done at the coarse-grain level. Since the electrostatic energy terms for a particular instance of a titratable site can be calculated independently from energy terms of other instances of sites, each instance is assigned a separate calculation thread. A similar approach is taken during the calculation of titration curves, where each pH-step of a curve is calculated separately.

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

```
cem.Initialize ()
```

During the initialization, the protein model is partitioned into titratable sites and a non-titratable background. Additionally, model compounds are generated. 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 next two lines generate a summary of the continuum electrostatic model and show a table of titratable sites:

```
cem.Summary ()
cem.SummarySites ()
```

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

```
cem.WriteJobFiles ()
```

By default, each site is assigned a separate directory, for example `scratch/PRTA/GLU8/`. Inside, there are PQR files for each instance of the site in the protein and in a model compound. The OGM and MGM files specify parameters of lattices used to solve the Poisson-Boltzmann equation. File `back.pqr` defines the non-titratable background and `protein.pqr` defines the whole protein and is used to calculate the boundary between the protein and the solvent. The boundary is calculated automatically by MEAD. The last file, `sites.fpt`, contains atomic coordinates and charges of all instances and is used for the calculation of interaction energies between different instances of sites in the protein.

5.3 Calculating electrostatic energy terms

At this point, the electrostatic energy terms can be calculated:

```
cem.CalculateElectrostaticEnergies ()
```

For each instance of each site, two electrostatic energy terms are calculated, namely the Born energy, G^{Born}, and the background energy, G^{back}. Born energy is the electrostatic energy of a set of charges interacting with its own reaction field. Background energy is the electrostatic energy of a set of charges interacting with other charges from outside of this set. The two energies are calculated for a particular instance of a site both in the model compound and in the protein. The difference $\left(G^{\mathrm{Born}}_{protein} + G^{\mathrm{back}}_{protein}\right) - \left(G^{\mathrm{Born}}_{model} + G^{\mathrm{back}}_{model}\right)$ is obtained, which is called the heterogeneous transfer energy, $G^{\mathrm{heterotrans}}$. The site is transferred from the model compound to the protein. In the model compound, the site has a model energy G^{model}, which corresponds to the experimentally known pK_{a} value of the deprotonation reaction in aqueous solution. The site in the protein has an intrinsic energy, $G^{\mathrm{intr}} = G^{\mathrm{model}} + G^{\mathrm{heterotrans}}$. Program `my_2diel_solver` calculates Born and background energies in the model compound. Program `my_3diel_solver` calculates

Born and background energies in the protein and, additionally, electrostatic interaction energies of the instance with all other instances of other sites in the protein. The toolkit collects Born, background and interaction energies from MEAD and calculates transfer and intrinsic energies.

Note that `my_3diel_solver` can in principle perform calculations in a three-dielectric environment (solvent, protein, vacuum). However, the electrostatic model employed here uses only two-dielectric environments, namely the solvent phase and protein/model compound phase.

5.4 Calculating microstate energies

After the G^{intr} values and interaction energies have been calculated and tabulated, one can calculate the energy of a particular protonation state of the protein, which is the so-called microstate energy, G^{micro}. The polypeptide in the "twosites" example contains only two sites, glutamate and histidine, so there can be $2^1 * 4^1 = 8$ possible protonation states, because glutamate has two instances ("p" and "d") and histidine has four instances ("HSP", "HSD", "HSE", "fully deprotonated"). For real-life proteins, the number of protonation states is very large. The protonation state of a protein is defined by a state vector. Each component of the state vector represents the protonation state of a site in the protein. The value of the component indicates the current instance of the site. The values of 0 and 1 do not necessarily mean "deprotonated" and "protonated". The next few lines of code generate all possible permutations of the state vector and calculate their respective microstate energies:

```
statevector = StateVector (cem)
increment   = True
while increment:
    Gmicro = cem.CalculateMicrostateEnergy (statevector, pH=7.0)
    statevector.Print (verbose=True, title="Gmicro = %f" % Gmicro)
    increment = statevector.Increment ()
```

5.5 Calculating probabilities of protonation states

In the *Pcetk* toolkit, the probabilities of protonation states can be calculated either analytically for small proteins or by Monte Carlo sampling for larger proteins. If a Monte Carlo model has not been defined, the default option is to calculate the probabilities analytically:

```
cem.CalculateProbabilities (pH=7)
cem.SummaryProbabilities ()
```

Argument `pH` can be omitted, since pH=7 is the default setting. For comparison, the probabilities can be calculated by using a Monte Carlo method. In the first step, a MC model is created and a number of 30000 production scans is defined. In the second step, the MC model is associated with the continuum electrostatic model by using the `DefineMCModel` method. From this moment on, the probabilities are estimated with Metropolis Monte Carlo.

```
mc = MCModelDefault (nprod=30000)
cem.DefineMCModel (mc)
cem.CalculateProbabilities ()
cem.SummaryProbabilities ()
```

The resulting tables show for each site the probability of occurrence of each instance. Instances with the highest probability of occurrence are marked with "*". Optionally, argument `reportOnlyUnusual=True` will only show sites in their non-standard protonation states, for example protonated glutamates.

5.6 Calculating titration curves

Titration curves are obtained by calculating protonation state probabilities for a given pH-range, usually from 0 to 14. The resolution of a curve can be adjusted with the `curveSampling` option (default 0.5 pH-unit).

```
from ContinuumElectrostatics import TitrationCurves
cmc = TitrationCurves (cem, curveSampling=0.5)
cmc.CalculateCurves ()
cmc.WriteCurves (directory="curves_mc")

cem.DefineMCModel (None)
ca = TitrationCurves (cem, curveSampling=0.5)
ca.CalculateCurves ()
ca.WriteCurves (directory="curves_analytic")
```

A `cmc` object is created representing titration curves. The curves are subsequently calculated and the results are written to files in `curves_mc/` directory. A simple two-column text file is written for each instance of each site that can be visualized in a plotting program of choice. Next, the Monte Carlo model is detached from the electrostatic model by passing a `None` argument. The curves are recalculated analytically and written to directory `curves_analytic/`.

5.7 Calculating microstate energies of substates

From the calculated probabilities, one can generate a state vector describing the lowest energy protonation state of the protein:

```
lowestEnergyVector = StateVector_FromProbabilities (cem)
```

The microstate energy, G^{micro}, in the lowest energy protonation state can be calculated in the following way:

```
Gmicro = cem.CalculateMicrostateEnergy (lowestEnergyVector)
```

By changing the values (=instances) of selected components (=sites) of the state vector, it is possible to calculate energies for a series of substates of the lowest energy state. Alternatively, substate energies can be calculated easier with the `MEADSubstate` class:

```
sites = (
    ("PRTA", "GLU" , 35),
    ("PRTA", "ASP" , 52), )
substate = MEADSubstate (cem, sites, pH=7.0)
substate.CalculateSubstateEnergies ()
substate.Summary ()
```

The above code was taken from the test case "lysozyme". First, a tuple of sites that make up the substate is defined. Second, a substate object is created and the lowest energy state vector is determined automatically by the toolkit at pH=7. Finally, state energies are calculated for the substate and a summarizing table is printed:

```
-------------------------------------------------------------
 State   Gmicro   Charge   Protons   PRTA GLU 35    PRTA ASP 52
-------------------------------------------------------------
     1      0.00       -2        0       d              d
     2      2.16       -1        1       p              d
     3      4.26       -1        1       d              p
     4      7.69        0        2       p              p
-------------------------------------------------------------
```

In the lowest energy state, Glu35 and Asp52 are deprotonated and the protonation of Glu35 by a proton coming from the solution requires about 2.2 $^{\text{kcal}}/_{\text{mol}}$.

6 Parameter files

Two parameter formats are recognized, YAML (pDynamo's default format) and EST (an extension of the ST format used by MEAD). File `parameters/radii.yaml` contains a list of atomic radii required to calculate the boundary between the protein and solvent phase. Files in the `parameters/sites/` directory define parameters for titratable sites. The listing below shows the parameter file for aspartate:

```
---
site      : ASP
atoms     : [CB, HB1, HB2, CG, OD1, OD2]
instances :
  - label   : p
    Gmodel  : -5.487135
    protons : 1
    charges : [-0.21,  0.09,  0.09,  0.75, -0.36, -0.36]

  - label   : d
    Gmodel  : 0.000000
    protons : 0
    charges : [-0.28,  0.09,  0.09,  0.62, -0.76, -0.76]
...
```

The site's name is defined by the parameter `site`. `atoms` defines a list of names of atoms that constitute the site. The consecutive lines define instances of the site. `label` is the name of an instance, for example "p" for "protonated". `Gmodel` is the energy of the model compound expressed in $^{\text{kcal}}/_{\text{mol}}$. The G^{model} value is derived from the aqueous solution pK_{a} value of aspartate, which is 4.0, calculated at 300 K. Model energies are recalculated if a different temperature is to be used. The reason for $G^{\text{model}} = 0$ for the deprotonated instance is that the energies are calculated relative to a reference state, which is in this case deprotonated aspartate. Parameter `protons` defines a number of protons bound to each protonation form (=instance) of aspartate. Finally, `charges` is a list of atomic charges that differ between the instances. The order of charges is the same as the order of atoms in the `atoms` list. The charges usually come from the CHARMM force field, i.e., they are parametrized for the dielectric constant $\epsilon = 4$.

Proteins often contain non-standard titratable sites, for example ligands or cofactors. Parameter files describing these sites are recognized by their extension and can be loaded automatically from the current directory. These additional parameter files can be provided both in the EST or YAML formats (the EST format takes precedence). If the names of the new sites coincide with the names of already existing sites, the parameters of the existing sites will be overwritten.

7 Future developments

- Extend the model by including rotameric instances
- Include reduction/oxidation reactions in addition to protonation/deprotonation reactions
- Develop a custom Poisson-Boltzmann solver to replace MEAD
- Optionally convert $^{\text{kcal}}/_{\text{mol}}$ (MEAD units) to $^{\text{kJ}}/_{\text{mol}}$ (pDynamo units)
- Make use of the MEAD program `pqr2SolvAccVol` to speed up calculations a bit

8 Feedback

The *Pcetk* toolkit and its documentation are in a stage of active development. Therefore, bugs are inevitable. If you found a bug, have a question or wish contribute to the code, please write to the developer: `mikolaj.feliks@gmail.com`.

9 References

[1] Bashford, D. *An Object-Oriented Programming Suite for Electrostatic Effects in Biological Molecules.* 1997, pp. 233–240.

[2] Bashford, D., and Gerwert, K. Electrostatic Calculations of the pK_a Values of Ionizable Groups in Bacteriorhodopsin. *J. Mol. Biol. 224* (1992), 473–486.

[3] Beroza, P., Fredkin, D. R., Okamura, M. Y., and Feher, G. Protonation of Interacting Residues in a Protein by a Monte Carlo Method: Application to Lysozyme and the Photosynthetic Reaction Center. *Proc. Natl. Acad. Sci. U.S.A. 88* (1991), 5804–5808.

[4] Bombarda, E., and Ullmann, G. M. pH-dependent pK_a Values in Proteins – A Theoretical Analysis of Protonation Energies with Practical Consequences for Enzymatic Reactions. *J. Phys. Chem. B 114* (2010), 1994–2003.

[5] Brooks, B. R., Bruccoleri, R. E., Olafson, B. D., States, D. J., Swaminathan, S., and Karplus, M. CHARMM: A Program for Macromolecular Energy, Minimization and Dynamics Calculations. *J. Comput. Chem. 4* (1983), 187–217.

[6] Essigke, T. *A Continuum Electrostatic Approach for Calculating the Binding Energetics of Multiple Ligands.* PhD thesis, University of Bayreuth, 2008.

[7] Field, M. J. The pDynamo Program for Molecular Simulations Using Hybrid Quantum Chemical and Molecular Mechanical Potentials. *J. Chem. Theory Comput. 4* (2008), 1151–1161.

[8] Humphrey, W., Dalke, A., and Schulten, K. VMD: Visual Molecular Dynamics. *J. Mol. Graph. 14* (1996), 33–38.

[9] Klingen, A., Bombarda, E., and Ullmann, G. Theoretical Investigation of the Behavior of Titratable Groups in Proteins. *Photochem. Photobiol. Sci. 5* (2006), 588–596.

[10] MacKerell, A. D. et al. All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins. *J. Phys. Chem. B 102* (1998), 3586–3616.

[11] Ullmann, G. M., and Knapp, E. W. Electrostatic Models for Computing Protonation and Redox Equilibria in Proteins. *Eur. Biophys. J. 28* (1999), 533–551.

[12] Ullmann, R. T., and Ullmann, G. M. GMCT: A Monte Carlo Simulation Package for Macromolecular Receptors. *J. Comput. Chem. 33* (2012), 887–900.

[13] Ullmann, T. *Monte Carlo Simulation Methods for Studying the Thermodynamics of Ligand Binding and Transfer Processes in Biomolecules*. PhD thesis, University of Bayreuth, 2012.

Websites

- pDynamo (Martin J. Field)

  `http://pdynamo.org`

- MEAD (Donald Bashford)

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

- Extended-MEAD (Thomas Ullmann)

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

- GMCT (Thomas Ullmann)

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