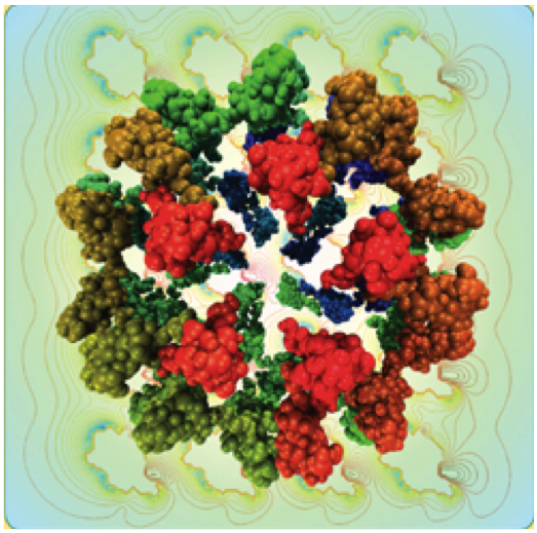
Reference Manual

Poisson Boltzmann Analytical Model (PB-AM)



Enghui-Yap
Albert Einstein College of Medicine

Lisa Felberg
Marielle Soniat
David Brookes
Teresa Head-Gordon
University of California, Berkeley

For more information, please visit http://thglab.berkeley.edu

Cover Illustration: An exploded view of a Brome Mosaic Virus capsid composed of T = 1 particles (PDB: 1YC6), represented as collections of overlapping spheres, is shown. PB-SAM is a new semi-analytical approach to efficiently solve the linearized Poisson-Boltzmann equation using multipole formalisms for overlapping spheres. The background shows the potential profile for an array of 601YC6 monomers computed using this method.

Recommended Citations:

When citing PB-AM in the literature, the following citation should be used
I. Lotan and T. Head-Gordon (2006). An analytical electrostatic model for salt screened interactions between multiple proteins J. Chem. Theory Comput 2, 541-555.

When citing PB-SAM in the literature, the following citations should be used
1. E.-H. Yap and T. Head-Gordon (2010). New and efficient Poisson-Boltzmann solver for interaction of multiple proteins J. Chem. Theory Comput. (Journal cover) 6, 2214-2224.
2. E.-H. Yap and T. Head-Gordon (2013). Calculating the bimolecular rate of protein?protein association with interacting crowders. J. Chem. Theory Comput. 9(5), 2481-2489.
3. O. N. Demerdash, E.-H. Yap and T. Head-Gordon (2014). Advanced potential energy surfaces for condensed phase simulation. Ann. Rev. Phys. Chem. 65, 149-174.

Acknowledgments: Research support from NIH and DOE is gratefully acknowledged.

Table of Contents

Chapter 1

Introduction

The Poisson-Boltzmann Analytical Model solves the linearized Poisson-Boltzmann Equation (PBE) for systems hitherto not possible using traditional PBE solvers. This manual describes the method and its associated suite of programs. The PBE software suite is licensed as a collection of freely available program under a GPL license.

1.1 PB-AM

The first general analytical solution for computing the screened electrostatic interaction between large numbers of macromolecules of arbitrarily complex charge distributions, assuming they are well described by spherical low dielectric cavities in a higher dielectric medium in the presence of a Debye-Hückel treatment of salt. The method exploits multipole expansion theory for the screened Coulomb potential such that it can describe direct charge-charge interactions and all higher-order cavity polarization effects between low dielectric spherical cavities containing their charges, while treating these higher order terms correctly at all separation distances. The analytical solution is general to arbitrary numbers of macromolecules, is efficient to compute, provides for the first time the ability to provide new benchmarks for other numerical solutions to the linearized Poisson-Boltzmann equation. A number of utilities are described below that use PB-AM results.

1.2 Physical Calculations

The PB-AM results allow for fast calculation of physically important quantities. Specifically this program can calculate the interaction energy of, and the force and torque applied to each molecule. These results may be written into a file or printed to the terminal.

1.3 Brownian Dynamics

This package also implements dynamics simulations using the Brownian protocol developed by Ermak and McCammon[1]. At each time step, the translation and rotation due to the applied force and torque, respectively, are calculated and added to random components of motion. The user can specify spatial and temporal termination conditions for the simulation and write the trajectory to specified files.

1.4 Electrostatics

Another possible output of PB-AM, the user can specify a configuration of an arbitrary number of molecules and get a 2-dimensional or 3 dimensional potential landscape of the system. We

[1]Ermak, D. L.; McCammon, J. A. *J. Chem. Phys.* 1978, *69*, 1352–1360.

have provided example plotting tools, and the 3-dimensional output may be uploaded in VMD for visualization.

Chapter 2

Theory

2.1 PB-AM formulation

PB-AM is an analytical solution to the linearized Poisson-Boltzmann equations for multiple spherical objects of arbitrary charge distribution in an ionic solution. The linearized Poisson-Boltzmann equation is given as:

$$\nabla[\epsilon(r)\nabla\phi(r)] - \epsilon(r)\kappa^2\phi(r) = 4\pi\rho(r)$$

$$\phi_{out}^{(i)} = \phi_{out}^{(i)}\Big|_{r=a_i}$$

$$\epsilon_s \frac{\partial\phi_{out}^{(i)}}{\partial r} = \epsilon_s \frac{\partial\phi_{out}^{(i)}}{\partial r}\Big|_{r=a_i}$$

Exploiting fast-multipole methods, this boundary value problem can be reduced to the following system of linear equations.

$$A = \Gamma \cdot (\Delta \cdot T \cdot A + E)$$

A(i) represents the effective multipole expansion of the charge distributions of molecule (i). E(i) is the free charge distribution of molecule (i). Γ is a dielectric boundary-crossing operator, Δ a cavity polarization operator, T an operator that transforms the multipole expansion to a local coordinate frame. More details on the method are available in Lotan, Head-Gordon (2006).

2.1.1 Physical Calculations

From the above formulation, computation of the interaction energies ($\Omega^{(i)}$) is given as follows:

$$\Omega^{(i)} = \frac{1}{\epsilon_s}\left\langle \sum_{j\neq i}^{N} T \cdot A^{(j)}, A^{(i)} \right\rangle$$

Where $\langle M, N\rangle$ denotes the inner product. When energy is computed, forces follow as:

$$\mathbf{F}^{(i)} = \nabla_i\Omega^{(i)} = \frac{1}{\epsilon_s}[\langle \nabla_i\, T \cdot A^{(i)}, A^{(i)}\rangle + \langle T \cdot A^{(i)}, \nabla_i\, A^{(i)}\rangle]$$

The method to calculate the torque $\boldsymbol{\tau}^{(i)}$ on molecule is outside the scope of this manual, but is discussed extensively in Lotan, Head-Gordon (2006).

2.1.2 Brownian Dynamics

Brownian dynamics simulations are implemented by treating each molecule as a Brownian particle experiencing a conservative force $\mathbf{F}^{(i)}$ and torque $\boldsymbol{\tau}^{(i)}$, as well as friction and random force due to the solvent. The translation Δr_i and rotation $\Delta\theta_i$ for a time step Δt are then given by

$$\Delta r^{(i)} = \frac{D_{i,trans}\Delta t}{k_B T}\mathbf{F}^{(i)} + \mathbf{S}_i(\Delta t)$$
$$\Delta\theta^{(i)} = \frac{D_{i,rot}\Delta t}{k_B T}\boldsymbol{\tau}^{(i)} + \boldsymbol{\Theta}_i(\Delta t)$$

where $D_{i,trans}$ and $D_{i,rot}$ are the translation and rotational diffusion coefficients for molecule i, respectively and $\mathbf{S}_i(\Delta t)$ and $\boldsymbol{\Theta}_i(\Delta t)$ are the stochastic components of translation and rotation, respectively, which have the following properties:

$$\langle \mathbf{S}_i \rangle = 0, \qquad \langle \mathbf{S}_i^2 \rangle = 2D_{i,trans}\Delta t$$
$$\langle \boldsymbol{\Theta}_i \rangle = 0, \qquad \langle \boldsymbol{\Theta}_i^2 \rangle = 2D_{i,rot}\Delta t$$

2.1.3 Electrostatics

In a similar manner to the interaction energy calculations in the Physical Calculations section, the electrostatics computes the potential at any point in space exterior to the molecules, $\mathbf{r}$, as the inner product of all the solved effective multipole expansions $A^{(j)}$ with a local multipole expansion of a single positive charge at position, $\mathbf{r}$.

$$\Phi_{out}(\mathbf{r}) = \frac{1}{\epsilon_s}\left\langle \sum_{j=1}^{N} T \cdot A^{(j)},\ A(\mathbf{r}) \right\rangle$$

Chapter 3

Use

3.1 Installation

To install this program, first download the source code from the GitHub repository:

```
$ git clone https://github.com/davas301/pb_solvers.git
```

Then navigate to the directory `pb_solvers/pbam`. From here, make a `build` and `bin` directory, use CMake to create a make file and then make the program. This is done as follows:

```
$ cd pb_solvers/pbam
$ mkdir build bin
$ cd build
$ cmake ..
$ make install
```

An executable named `pbam` will now be in `pb_solvers/bin`.

3.2 Input Information

Input Files

The program executable requires an input file as a command line parameter. The input file contains the various arguments and parameters that one may wish to set when running the program. Each line of the input file contains a keyword followed by a variable number of whitespace-delimited parameters, e.g.:

```
keyword1    param1    param2
keyword2    param1    param2    param3
```

Each keyword is described in the table below, along with its associated parameters.

Keyword	Parameters	Description
runname	<name>	<name> is desired internal name of this run.
attypes	<numtypes>	Set the number of different atom types to <numtypes>.
pqr	<idx> <fpath>	The molecule index for this xyz file. Provide input PQR file at <fpath>.
xyz	<idx> <fpath>	The molecule index for this xyz file. Provide input XYZ file at <fpath>.
randorient		If you want your molecules to be randomly rotated, use this flag
units	<units>	Set the units of output to <units>. The current options are: jmol (Joules/mole), kT and kcalmol (kCal/mole).
salt	<con>	Set salt concentration in the system to <con>.
temp	<T>	Set system temperature to <T>
idiel	<ival>	Set the interior dielectric constant to <ival>.
sdiel	<sval>	Set the solvent dielectric constant to <sval>.
pbc	<boxlength>	Set size of periodic box to <boxlength>.
random	<seed>	Seed the internal random number generator with <seed>.
type	<idx> <ct> <movetype> <dtr> <drot>	Set attributes of an atom type, where <idx> is the integer id of this type, which can be 1 to <numtypes> (above). <ct> is the number of atoms of this type in the system and <movetype> describes the way this type is allowed to move in a dynamics run (move, rot, or stat). If <movetype> is move, then a translational diffusion coefficient <dtr> and a rotational diffusional coefficient <drot> are required. If <movetype> is rot then just <drot> is required.

Keyword	Parameters	Description
runtype electrostatics	<gridpts>	Will run electrostatics calculations. <gridpts> is an optional integer describing the number of evenly spaced points in each dimension to perform calculations on.
dx	<fname>	For electrostatics. Will write the results of electrostatics calculations for every 3D grid point to <fname>, in the same output format as APBS dx file.
gridct	<ct>	For electrostatics. <ct> is the number of 2D grids to output.
grid2d	<idx> <fname> <axis> <val>	For electrostatics. Set attributes of a grid output where <idx> is the integer id of this grid, which can be 1 to <ct> (above). Will write output of calculations for a cross section along <axis> (x, y, or z) at <value>.
runtype dynamics	<outname> <ntraj>	Will perform a brownian dynamics run. A directory where trajectory information will be stored in and the number of trajectories is required.
termct	<ct>	Set number of termination conditions to <ct>.
termcombine	<andor>	Set how termination conditions will be combined. <andor> should be and or or. Default is or.
term	<idx> <type> <val> <mols>	Set attributes of a termination condition where <idx> is the integer id of this condition, which can be 1 to <ct> (above). <type> can be time, contact, x<=, y<=, z<=, or r<= (or the >= equivalents of these), <val> is the value where the simulation will terminate and <mols> is a whitespace-delimited list of molecular indices that this condition applies to (contact requires 2, time requires 0, and all else require 1).
xyz	<idx> <trajidx> <fpath>	<idx> is the molecule index for this xyz file. Provide input XYZ file at <fpath>. For the dynamics run, a starting configuration is needed for each trajectory for all the molecule types, so there should be <ntraj> xyz lines for each molecule, the trajectory number denoted by <trajidx>.
runtype energyforce	<outfilename>	Will calculate the interaction energy, the forces and torques for the system input. <outfilename> is a filename that you would like the information printed to. If none is entered, the information will be printed to the command line.

System inputs: PQR and XYZ Files

All the options above require a **PQR** file name. A PQR file can be generated from a PDB file using the PDB2PQR program, available as a web server or for download at:

http://nbcr-222.ucsd.edu/pdb2pqr_1.9.0/
http://www.poissonboltzmann.org/docs/pdb2pqr-installation/

It may also be formatted manually. The general format of a PQR file is as follows, and is whitespace-delimited:

```
recName serial atName resName chainID resNum X Y Z charge rad
```

Parameter	Description
`recName`	A string that should either be ATOM or HETATM.
`serial`	An integer that provides the atom index
`atName`	A string that provides the atom name.
`resName`	A string that provides the residue name.
`chainID`	An optional string that provides the chain ID of the atom.
`residueNumber`	An integer that provides the residue index.
`X Y Z`	Three floats that provide the atomic coordinates.
`charge`	A float that provides the atomic charge (in electrons).
`Rad`	A float that provides the atomic radius (in Å).

The **XYZ** file simply specifies the desired molecule centers for a given molecule type.

```
mol1X mol1Y mol1Z
mol2X mol2Y mol2Z
mol3X mol3Y mol3Z
```

Chapter 4

Example files and outputs

4.1 Physical calculations

Example physical calculations runfile

name: `run.energyforce.inp`:

```
runtype energyforce
runname energyforce.2sp.jmol.out

units jmol
salt 0.01
temp 353
idiel 4
sdiel 78

attypes 1
type 1 2
pqr 1 single_charge.pqr
xyz 1 positions_2.xyz
```

The files for PQR and XYZ are:

name: `single_charge.pqr`:

```
ATOM      1  N   NTR     0       1.000   0.000   0.000 -1.0000  3.7300
ATOM      1  N   NTR     0       0.000   1.000   0.000 -1.0000  6.3200
```

name: `positions_2.xyz`:

```
-10.0   23.4   -8.7
  0.0    0.0   -2.5
```

To run:

```
$$ ../../bin/pbam run.energyforce.inp
```

And the resulting file:

name: `energyforce.2sp.jmol.out`:

```
My units are Joules/Mol
MOLECULE #1
        POSITION: [-10, 23.4, -8.7]
        ENERGY: 1328.86
        FORCE: 1.19858e+08, [-36.6012 1.19858e+08 -1.65408e-05]
        TORQUE: 1.78426e+06, [1.28425 1.78426e+06 1.9978e-06]
MOLECULE #2
        POSITION: [0, 0, -2.5]
        ENERGY: 1328.86
```

```
        FORCE: 1.19858e+08, [36.6012 -1.19858e+08 1.65408e-05]
        TORQUE: 1.78354e+06, [1.28372 1.78354e+06 1.99699e-06]
```

4.2 Brownian Dynamics

4.3 Electrostatics

Example electrostatics runfile

name: `run.electrostatic.inp`:

```
runtype electrostatics 140
runname electrostatic

units kT
salt 0.01
temp 298
idiel 4
sdiel 78

dx out.dx

gridct 2
grid2D 1 out.x.0.dat x 0
grid2D 2 out.x.-1.dat x -1

attypes 2
type 1 2
pqr 1 single_charge.pqr
xyz 1 positions_2.xyz

type 2 2
pqr 2 pos_charge.pqr
xyz 2 positions_pos.xyz
```

The files for PQR and XYZ files are:

name: `single_charge.pqr`:

```
ATOM      1  N   NTR     0       0.000   1.000   0.000  4.0000  0.3200
ATOM      1  N   NTR     0       0.000   0.000  -1.000  4.0000  0.3200
ATOM      1  X   CEN     0       0.000   0.000   0.000  0.0000  2.0000
```

name: `positions_2.xyz`:

```
  0.0   0.0  -5.0
  0.0   0.0   5.0
```

name: `pos_charge.pqr`:

```
ATOM      1  N   NTR     0       0.000   1.000   0.000 -4.0000  0.3200
ATOM      1  N   NTR     0       0.000   0.000  -1.000 -4.0000  0.3200
ATOM      1  X   CEN     0       0.000   0.000   0.000  0.0000  2.0000
```

name: `positions_pos.xyz`:

```
  0.0   5.0   0.0
  0.0  -5.0   0.0
```

To run:

```
$$ ../../bin/pbam run.electrostatic.inp
```

And the resulting files:

name: `out.dx`:

```
# Data from PBAM Electrostat run
# My runname is out.dx and units kT
object 1 class gridpositions counts 140 140 140
origin -4 -9 -9
delta 0.0571429 0.0e+00 0.0e+00
delta 0.0e00 0.128571 0.0e+00
delta 0.0e00 0.0e+00 0.128571
object 2 class gridconnections counts 140 140 140
object 3 class array type double rank 0 items 2744000 data follows
2.7203115e-01  3.0271755e-01  3.3459723e-01
3.6769040e-01  4.0201595e-01  4.3759129e-01
.....
-1.3185519e-01  -1.5849252e-01  -1.8359631e-01
-2.0722087e-01  -2.2942006e-01  -2.5024714e-01
-2.6975467e-01  -2.8799442e-01
attribute "dep" string "positions"
object "regular positions regular connections" class field
component "positions" value 1
component "connections" value 2
component "data" value 3
```

name: `out.x.0.dat`:

```
# Data from PBAM Electrostat run
# My runname is out.x.0.dat
units kT
grid 140 140
axis x 0
origin -9 -9
delta 0.128571 0.128571
maxmin 39.23 -39.23
    0.3605004      0.4030045      0.4474874      0.4940082      0.5426260      0.5933995
```

Chapter 5

Common Errors

5.1 Segmentation Fault while reading input file

Our file reader is not very robust. Make sure there is only a single space between keywords and options. Make sure that all options are specified for a given keyword.

5.2 Initial configuration errors

Many issues may arise if your molecules are overlapping, which is not allowed in the PB solvers' models. The center of the molecule will be placed at the positions given by each XYZ file. Check the printed out PQR file for the initial configuration, which you can load into VMD. Try changing the xyz file(s). This error may also appear if the box length specified with the pbc keyword is too small. Try increasing the box length.

Chapter 6

Analysis Tools

6.1 Viewing PQR in VMD

Load pqr file

```
>> set selall [atomselect top "all"]
```

To center the pqr

```
>> $selall moveby {-x -y -z}
```

where x, y, and z are half the box length
Graphical representations: To view the charges inside the CG center, from the toolbar, select Graphics > Representations. In the selected atoms, type

```
not name X
```

Change the coloring method to Charge, and the Drawing Method to VDW. Then select the Create Rep button, and in the selected atoms, type

```
not name X
```

Change the Drawing Method to VDW and the Material to Transparent. The Graphical Representations and final images are in figure below.

6.2 Viewing Electrostatics in VMD

To load electrostatic results: File > New Molecule. In the window that appears, toggle the *Load files for:* to select the currently loaded PQR file. Then select *Browse* to find the location of the dx file. Once found, hit the *Load* button and let the dx file load.

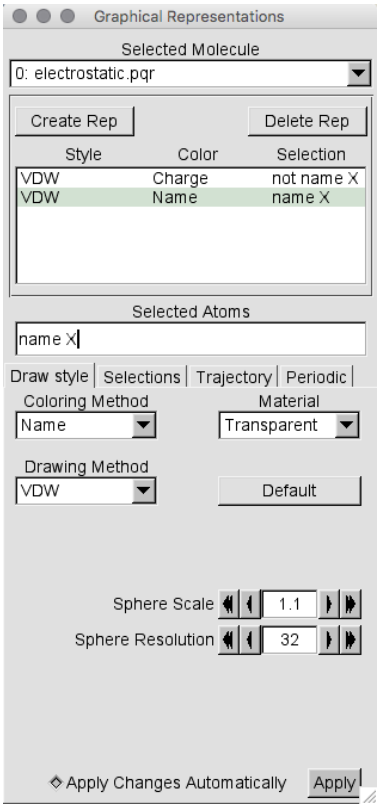


Figure 6.1: Graphics

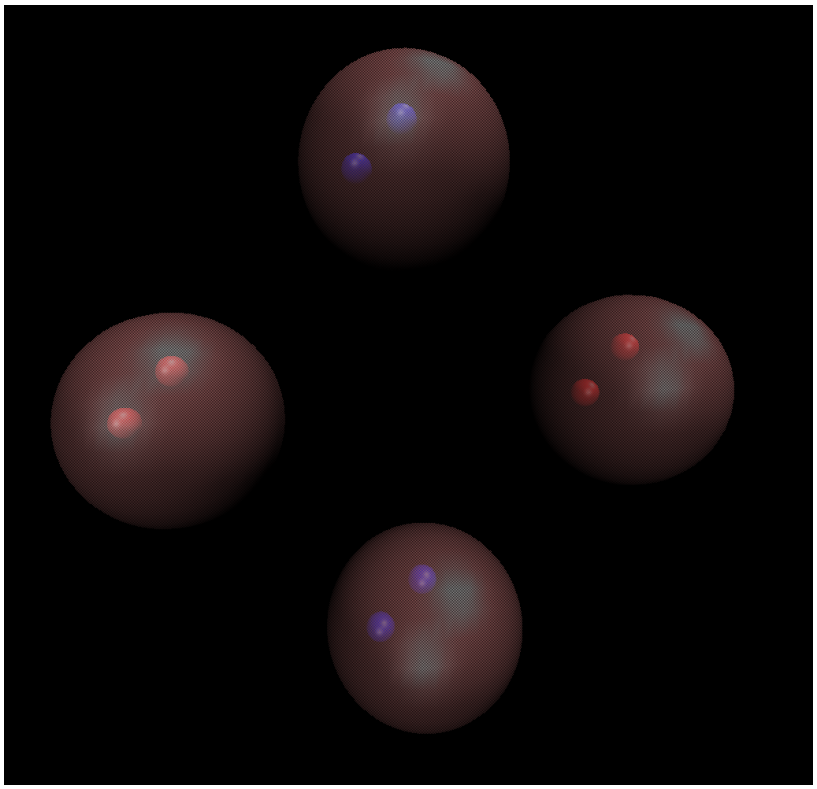


Figure 6.2: Final view

6.3 ESP plots

run with pot option in BD input file with a file which contains a list of z positions at which to calculate ESP (only capable of xy cross section at the moment) copy .out and esp_plot.py to Desktop Change filename within .py file, may need to adjust vmin and vmax in .py for color scale

```
>> python esp_plot.py
```

creates .pdf

To do: automatically read in trajectory and calculate ESP at those points cross-section through xz and yz planes ESP on the spheres' surfaces