User guide for GHOAT.py – v1.0

Table of Contents

1. Introduction

The Guest-HOst Affinity Tool (GHOAT.py) is a python tool designed to fully automate absolute binding free energy calculations on guest-host (or host-guest) systems, starting only from an initial structure. It supports the *pmemd.cuda* sofware from AMBER 20 or later [1], or alternatively the OpenMM simulation engine combined with OpenMMtools [2,3], both exhibiting high performance on Graphics Processing Units (GPUs). In addition to their role as catalysts, guest-host systems are important since they provide small test systems for binding free energy calculations, which can be employed for parameter evaluation and optimization [4].

In this user guide we will first describe the theory and methodology of GHOAT.py, with the simultaneous decoupling and recoupling (SDR) approach combined with the application/removal of restraints on the guest and host. We will then explain how the equilibrium simulations and free energy calculations are performed, and how they are analyzed in order to obtain the quantities of interest. All the parameters needed for the program input file, and how they apply to the various calculation steps, will also be described in detail. Finally, we will explain how to add a new host to our automated protocol, in addition to the ones provided by the GHOAT.py distribution.

2. Theory and methods

The expression for the calculated binding free energy for the SDR method is is defined as follows [5]:

$$-\Delta G^{o}_{bind} = \Delta G_{h,att} + \Delta G_{g,conf,att} + \Delta G_{g,TR,att} + \Delta G_{SDR} + \Delta G_{g,TR,rel} + \Delta G_{g,conf,rel} + \Delta G_{h,rel} \qquad (1)$$

In the equation above, the *att* index denotes attachment of restraints in the bound state, and *rel* indicates release of restraints with the guest and host separated, both in bulk solvent. The *h* and *g* indexes are for

host and guest, respectively, *conf* is for conformational restraints and *TR* is for translational/rotational restraints. The ΔG_{SDR} term is the free energy of transferring the ligand (guest) from the receptor (host) binding site to bulk with all restraints applied, using the SDR method. Each of these free energy components will be calculated using a series of simulations, as explained below.

2.1 Restraint setup

As shown above, the applied restraints can either be conformational (*conf*), meaning that they are applied to atoms belonging to the same molecule (host or guest), or translational/rotational (*TR*), which are restraints on the guest relative to the host.

The conformational restraints for both the guest and the host use the same procedure as in reference [5], with harmonic restraints applied on all non-hydrogen dihedrals of a given molecule, and they are optional. For the host, there is the alternative of applying only distance restraints between its anchors, by choosing it in the GHOAT input file (variable `host_rest_type` in section 5). The conformational restraints are applied/released in the two ends of the calculation, and are designed to limit the conformational freedom of the host and guest during the SDR process. Their contribution to the final binding free energy is calculated using a number of simulation windows with intermediate values of the harmonic spring constants [5,6], and the result is processed using the Multistate Bennett Acceptance Ratio (MBAR) method [7]. For the attaching process, the procedure is applied to the guest-host complex, and the restraints release is performed with the two molecules in separate boxes.

The *TR* restraints of the guest relative to the host use three anchor atoms in the guest and three in the host, being applied to one distance, two angles and three dihedrals formed between them (left of Figure 1). They are first applied to the bound system with the chosen conformational restraints fully attached, using a series of windows and MBAR to retrieve the *TR* attach free energy, as done in the conformational case. For their release, the following analytical expression is used [5,6]:

$$\Delta G_{g,TR,rel} = k_B T \ln\left(\frac{C^o}{8\pi^2}\right) + k_B T \ln \int_0^{2\pi}\int_0^{\pi}\int_0^{\infty} \exp[-\beta(u_r + u_\theta + u_\phi)] r^2 \, dr \sin\theta \, d\theta \, d\phi$$

$$+ k_B T \ln \int_0^{2\pi}\int_0^{2\pi}\int_0^{\pi} \exp[-\beta(u_\Theta + u_\Phi + u_\Psi)] \sin\Theta \, d\Theta \, d\Phi \, d\Psi \qquad (2)$$

Here C° is the standard concentration, 1 M = 1/1661Å^3, and r, θ and ϕ are the distance between the H1 and G1 anchors (H1-G1), angle H2-H1-G1, and H3-H2-H1-G1 dihedral, respectively. In the last term on the right, which integrates over guest orientation relative to the host, Θ is the angle H1-G1-G2, Φ is the dihedral H2-H1-G1-G2, and Ψ is the dihedral H1-G1-G2-G3. The u terms are the potential energies from the harmonic restraints, defined as $u = k\,(x - x_0)^2$, with x being a given coordinate with its reference value x_0, and k the spring constant. The ½ term is omitted following the AMBER definition of harmonic restraints between single atoms.

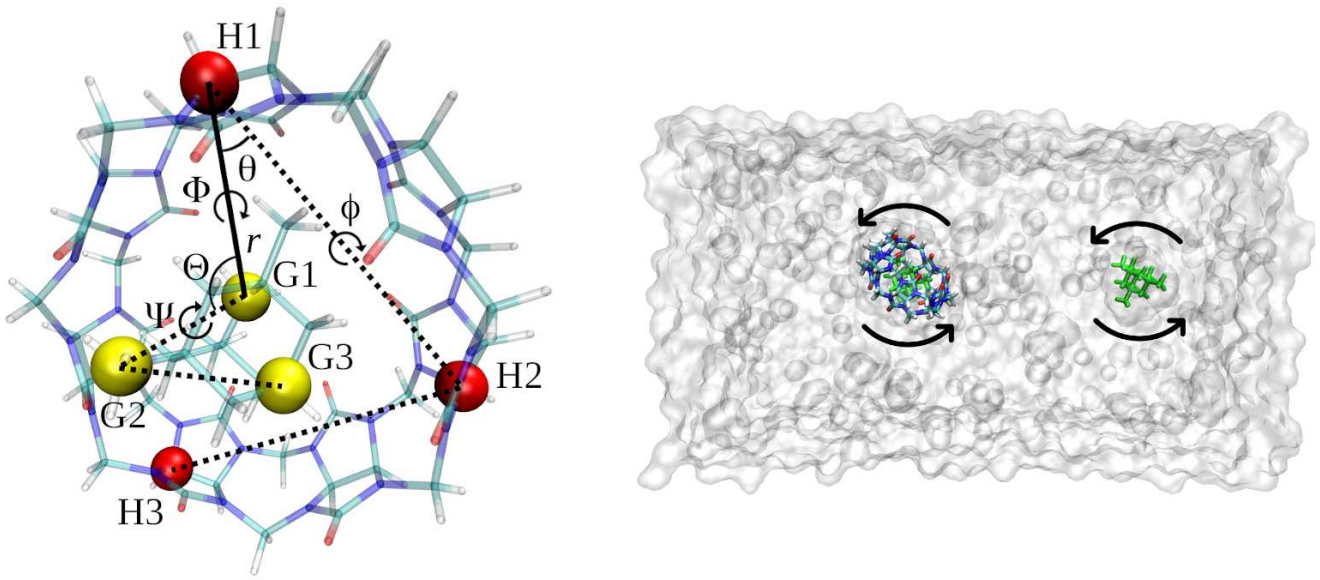

Figure 1: (left) Restraint scheme, showing the anchor atoms and the restrained coordinates. (right) The SDR procedure, with the complex and free guest in the same box, with applied COM restraints that still allow rotation.

2.2 Alignment and anchor atom assignment

The host anchors H1, H2 and H3 are predetermined for a particular host, and have to be included in the GHOAT input file. Instructions on how to assign the anchors for a new host, as well as other host parameters, are shown in section 6. The automatic assignment of the guest anchors G1, G2 and G3 follows a few rules and is done in the beginning of the equilibrium stage, as well as in the beginning of the free energy calculations, the latter always starting from the respective equilibrated structure. This procedure is explained in the paragraphs below.

Starting from the initial or the equilibrated structure, the orient plugin from VMD [8] will align the host's symmetry axis with the *z* axis, through the calculation of the hosts' three principal moments of inertia. After this, the origin of the system of coordinates will be placed at the center of mass of the host's non-hydrogen atoms (left of Figure 2). Starting from this aligned system, G1 anchor candidates will be searched inside a cylinder aligned with the *z* axis and with equal height and diameter (right of Figure 2), both chosen in the input file using the variable `l1_range` (section 5). If no guest atoms can be found inside this cylinder, the guest is considered to have left the binding site and the system is not built. From the candidate G1 atoms found inside the cylinder, G1 will be the one with the smallest value of $r_1^2 = (x_1^2+y_1^2)$, with x_1 and y_1 being its coordinates in the *x* and *y* axes.

The choice of G2 is made so that the Ψ dihedral from Figures 1 and 2 is aligned (or nearly aligned) with the axis of symmetry of the host (*z*). Thus, G2 will be the atom with the G1-G2 distance inside the anchor atom distance range, and having the smallest value of $r_d^2 = (x_d^2+y_d^2)$, with x_d and y_d being the projection of the G1-G2 distance along the *x* and *y* axes, respectively. The anchor atom distance range is chosen in the input file using the `min_adis` and `max_adis` variables. The G3 atom will be the one with the G1-G2-G3 angle closest to 90 degrees, and with the G2-G3 distance also falling within the specified distance between anchors. Choosing minimum distances between anchors, as well as angles close to 90 degrees, avoid crashes in the simulation due to the application of large forces on dihedral restraints caused by a gimbal lock.

The motivation behind the choices of G1 and G2 is to define a restrained degree of freedom (the Ψ dihedral) that reflects the rotation of the guest around the host's symmetry axis (right of Figure 2). This setup allows the user to leave this dihedral free during the SDR process and the *TR* restraints

attach/release procedures, by setting the `guest_rot` variable to "yes" in the input file. In that case, the free energies of attaching and releasing the Ψ dihedral restraint will be zero, with the contribution of this rotation to the binding free energy coming from the sampling of states during the SDR process. This could avoid the need of analytical corrections, due to the symmetry of the host producing identical states for a set of values of the Ψ dihedral, even though the Ψ restraint confines it to a single value.

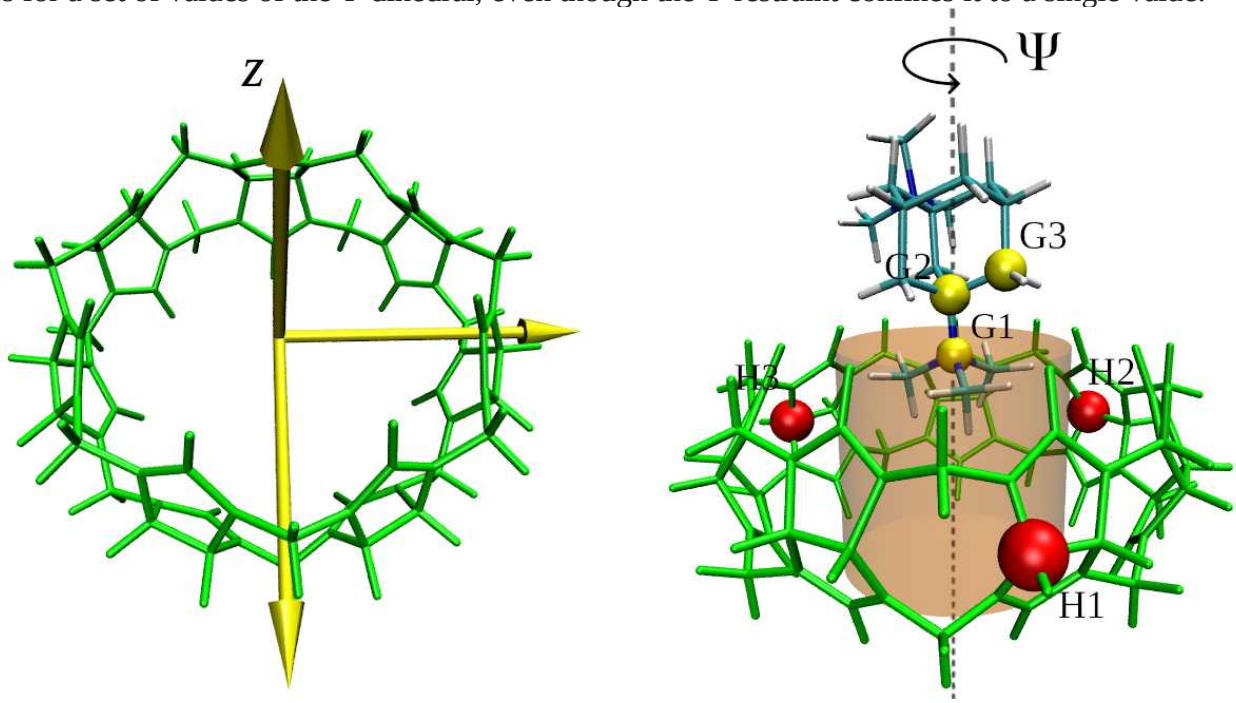

Figure 2: (left) The three principal moments of inertia of the host, with the symmetry axis along the *z* axis (right) Choice of the guest anchor atoms, with the Ψ rotation around a *z* axis passing through the host's center of mass.

2.3 SDR procedure

The SDR method has been described previously in the BAT.py code [5,6], so here we go over its general aspects, as well as the changes made for the GHOAT code. The simultaneous decoupling and recoupling process decouples a restrained guest from the rest of the system when the guest is bound to the host, and at the same time recouples a restrained guest in the same system, but away from the complex and considered in bulk solvent (right of Figure 1). This simultaneous process allows for the computation of the binding free energy of ligands that carry a net charge, without the need for analytical corrections, which would be required if the decoupling/recoupling happened in separate boxes [9].

In order to keep the guest-host complex and the bulk guest away from each other during the SDR calculations, center of mass (COM) restraints are applied to all non-hydrogen atoms of the host molecule from the complex. The same way, COM restraints are applied to all non-hydrogen atoms of the guest molecule that is located in bulk solvent. This ensures that the conformational spaces of both molecules are not affected by the COM restraints, even though this space is already limited if *conf* restraints are used. Note that both molecules are still allowed to rotate around their center of mass, which does not affect the calculations, as long as they don't get too close together (right of Figure 1). The distance between them can be optimally chosen in the input file, as explained in section 5.

The SDR calculation is performed separately for the decoupling/recoupling of the electrostatic (Eq. 4) and Lennard-Jones (Eq. 5) components of the guest-host interactions, using the expressions:

$$\Delta G_{SDR} = \Delta G_{elec} + \Delta G_{LJ} \quad , \tag{3}$$

$$\Delta G_{elec} = \Delta G_{elec,bound} - \Delta G_{elec,unbound} \quad (4)$$

$$\Delta G_{LJ} = \Delta G_{LJ,bound} - \Delta G_{LJ,unbound} \quad , \quad (5)$$

where the subscript *LJ* denotes Lennard-Jones interactions, *elect* the electrostatic interactions, and *bound* and *unbound* the state of the guest relative to the host. The calculation of these free energy contributions takes place through a series of simulation windows, with the final free energy value being computed using either Thermodynamic Integration with Gaussian Quadrature (TI-GQ), or the MBAR method as with the restraint calculations [5,6].

3. Equilibration simulations

Starting from an initial structure of the guest-host complex, GHOAT prepares the system for the equilibration step, so that the free energy step from the next section will (hopefully) start from a free energy minimum. The necessary parameters are also generated at this stage, if they are not already provided by the user. GHOAT is able to use the General Amber Force-Field (GAFF or GAFF2) [10,11] parameters for the bonded and LJ interactions, and employs the AM1-BCC [12] charge model for the partial charges of the guest and host.

With the system prepared, a series of equilibration simulations is then performed, in which the restraints on the guest relative to the host are gradually released, followed by a final unrestrained simulation. The final state of the complex after this procedure will be the reference state for all the free energy calculations below.

4. Free Energy Components

As with the BAT.py software, each free energy component from Equations 1 and 3 is identified by a letter, as shown in Table 1.

Table I: Binding free energy components, with the associated system, free energy method and contribution.

Description	Letter	System	Free Energy Method	Free energy term
Attachment of host conformational restraints	**a**	Complex	MBAR	$\Delta G_{h,att}$
Attachment of guest conformational restraints	**l**	Complex	MBAR	$\Delta G_{g,conf,att}$
Attachment of guest TR restraints	**t**	Complex	MBAR	$\Delta G_{g,TR,att}$
Simultaneous dec/recoupling of guest charge interactions	**e**	Complex + bulk guest	MBAR/TI	ΔG_{elect}
Simultaneous dec/recoupling of guest LJ interactions	**v**	Complex + bulk guest	MBAR/TI	ΔG_{LJ}
Release of guest TR restraints	**b**	Guest only	Analytical	$\Delta G_{g,TR,rel}$
Release of guest conformational restraints	**c**	Guest only	MBAR	$\Delta G_{g,conf,rel}$
Release of host conformational restraints	**r**	Host only	MBAR	$\Delta G_{h,rel}$

When the calculations are set up, the windows from each free energy component will be in folders named according to their corresponding letter followed by the window number, starting at *00*. When the MBAR method is used with OpenMM, all simulation windows from a given component are performed simultaneously with Hamiltonian Replica Exchange, and will be in the same folder identified by its component letter (ex: *e-comp* folder). The number of windows and their properties can be defined in the input file. The letters also identify the free energy output files, which for AMBER are stored in the ./data folder of each component after the analysis is performed.

5. Input file

5.1 Variables for use with *pmemd.cuda*

Various options concerning the creation of the systems, simulations and analysis, can be chosen in the input file:

`host:` The name of the host, which has to match the naming of the initial complex structures. For example, for the host named host-cb7, the initial pdb structure should be called host-cb7-<guestname>.pdb, with the <guestname> section explained below.

`guest_list:` The list of guests names that will be used for the calculations on one particular host. The list should be placed in brackets ans separated by commas. Ex: “[guest-1,guest-2,guest-3]”. Each item of the list corresponds to the <guestname> string in the initial structure file, so for host-cb7 and guest-1 calculation, the initial pdb file of the complex should be called host-cb7-guest-1.pdb.

`host_code:` The three letter residue identifier for the host molecule, which defines the residue(s) name of this molecule (Ex: CB7).

`guest_list_code:` The same as `host_code`, but for the residue three letter codes of the guests. Should correspond to the respective molecule in the `guest_list` array, so if the latter is “[guest-1,guest-2,guest-3]”, `guest_list_code` should be “[ML1,ML2,ML3]” if guest-1 is called ML1, guest-2 ML2 and so on.

`H1, H2` and `H3:` These define the anchor atoms of the host, which have to be determined beforehand, using AMBER masks to define each atom. Ex: “:1@C1” for the C1 atom of the first residue of the host.

`final_host_num`: Final residue number of the host, or the total number of residues of the host, since they have to start at 1 in the input structure. Only needed for hosts with multiple residues, such as cyclodextrins. Default value is 1 (single residue host).

`fe_type`: Type of binding free energy calculation. If SDR with restraints will be performed, choose “all”. For only the SDR components without computing the free energy of attaching/releasing restraints, choose “sdr”, or “rest” for restraints only. One can also choose the option “custom”, for a chosen set of components (see below).

`components`: If the option “custom” is set in the option above, choose the components you want to calculate, using a list of letters separated by spaces inside a bracket. Ex: “[l t e v c]”.

`sdr_dist`**:** Distance (in Å) between the bound guest and the copy of the guest located in bulk solvent (measured along the *z* axis), as required for the SDR method. The value of this variable should be large enough that the interactions of the complex with the bulk copy of the guest are negligible.

`release_eq`: The weights for the gradual release of the restraints in the equilibrium stage, going from 100 (fully restrained) to 0 (unrestrained). Each option will be a new simulation, and they are performed in sequence. Use a list of letters separated by spaces inside a bracket to define these weights. Ex: “[5.00 2.50 1.00 0.00]”. A single 0.00 inside the brackets (Ex: “[0.00]”) will run just one equilibrium simulation without any restraints.

`attach_rest`: List of weights as above, but for the spring constant of each window during the attaching/releasing of restraints using MBAR (components **a**, **l**, **t**, **c** and **r**). The total number of windows for each of these components will be the size of the array. Ex: “[0.00 2.00 4.00 16.00 64.00 100.00]” for a total of 6 windows. See refs [5,6] for more information.

`lambdas`: Lambda values for the decoupling procedure using SDR, going from 0.00 to 1.00. Only used with the MBAR method, since the lambda values are determined automatically when using TI-GQ (see `dec_int` and `ti_points` variables below).

`dec_int`: Type of integration method for the SDR components of the binding free energy calculation (**e** and **v**). If “TI” is chosen, Gaussian quadrature is applied, if “MBAR” is chosen, the latter is used to calculate these components.

`ti_points`**:** Number of points for Thermodynamic Integration with Gaussian Quadrature (TI-GQ), which will determine the lambda values and Gaussian weights when using this method. Accepts any positive integer.

`rec_dihcf_force`: Final spring constant for the host conformational dihedral restraints, as explained in section 2.1.

`rec_discf_force`: Final spring constant for the host conformational distance restraints, as explained in section 2.1.

`lig_dihcf_force`: Final spring constant for the guest conformational dihedral restraints, as explained in section 2.1.

`lig_distance_force`: Force constant for the *r* distance (H1-G1) of the *TR* restraints on the guest relative to the host, as explained in section 2.1 and shown in Figure 1.

`lig_angle_force`: Force constant for the angle/dihedral *TR* restraints on the guest relative to the host, as explained in section 2.1 and shown in Figure 1.

`rec_com_force`**:** Force constant for the center of mass restraints on the host, as explained in

section 2.3.

`lig_com_force`: Force constant for the center of mass restraints on the bulk guest during the SDR procedure, as explained in section 2.3.

`guest_rot`: Allow rotation of the restrained guest along the host symmetry axis (Ψ dihedral), as explained in section 2.2. Default is “no”, also accepts “yes”.

`host_rest_type`: Use non-hydrogen dihedrals or anchor distance conformational restraints for the host, as explained in section 2.1.

`water_model`: The water model used in the calculations. Supported options are “TIP3P”, “TIP4PEW” and “SPCE”.

`num_waters`: Number of waters used in the simulations of the complex (including SDR) and the free host box.

`buffer_x`, `buffer_y` and `buffer_z`: Options for the water padding in the three Cartesian axes of the system. The `buffer_z` option is mutually exclusive with the `num_waters` option above, since for a fixed number of waters the *z* padding is a dependent variable, and vice-versa.

`lig_buffer`: Water padding in the three Cartesian axes for the box with only the guest in it.

`neutralize_only`: Option to add ions only to neutralize the system, or to also include an additional number of ions. Accepts options “yes” or “no”.

`cation` and `anion`: Cation and anion species to be used, accepts all ions supported by the Joung and Cheatham monovalent ion parameters [13]. Ex: “Na+” and “Cl-”.

`ion_conc`: Salt concentration of the chosen ions for all simulation boxes, when additional ions are included after the system neutralization. Use units of mol/L. (Ex. 0.15).

`hmr`: Use hydrogen mass repartitioning [14] or not. Accepts options “yes” and “no”.

`temperature`: Temperature of all simulated systems, in Kelvin (K).

`eq_steps1`: Number of steps for each simulation of the gradual release of restraints, during the equilibration procedure.

`eq_steps2`: Number of steps for the last simulation of the equilibration procedure, in which the guest is unrestrained.

`[component]_steps1`: Number of steps of equilibration, for each window of the various components of the free energy calculation, with the component letters shown in Table I. No data is collected during this simulation.

`[component]_steps2`: Number of steps for the production stage of each window of the various components of the free energy calculation, in which data is collected.

`l1_range`: Diameter and height of the cylinder used in the search range for the first guest anchor G1, centered on the center of mass of the host (see section 2.2).

`min_adis` and `max_adis`: Minimum and maximum distance between the guest anchors.

`blocks`: Number of blocks for block data analysis. This separates the simulation data in blocks and provides the results for each, so the temporal variation and convergence of the results can be assessed. The standard deviation across the blocks is used for the calculation of the uncertainties of each free energy component.

`ntpr`, `ntwr`, `ntwe`, `ntwx`, `cut`, `gamma_ln`, `barostat` and `dt`: Options for running the various simulations, such as output frequency, non-bonded cutoff, barostat type, time step, and others. These use the same variables as the ones from the *pmemd.cuda* simulation input file, and their definitions can be found in the AMBER user guide.

`guest_list_charge`: Net charge of the guests, in case charged parameters are not provided. Should correspond to the respective molecule in the `guest_list` array, so if the latter is "[guest-1,guest-3]", `guest_list_charge` should be "[1,2]" if guest-1 has net charge +1 and guest-3 has net charge +2.

`amber_ff`: Choice of force field for the host and guest Lennard-Jones and bonded parameters, if not already provided. Accepts either "gaff" or "gaff2".

5.2 Specific variables for use with OpenMM

When using GHOAT with the OpenMM software, based on Ref. [6], a few of the AMBER variables from the section above are not needed, as shown in the example input files provided in the tutorial. In addition, the OpenMM simulations/calculations require a few OpenMM-specific variables, which are listed below.

`[component]_itera1`: Number of replica exchange equilibration iterations for each window from a given component of the binding free energy calculation. Replaces the `[component]_steps1` variable from the AMBER version.

`[component]_itera2`: Number of replica exchange production iterations for each window from a given component of the binding free energy calculation. Replaces the `[component]_steps2` variable from the AMBER version.

`dlambda`: Size of the δlambda interval for the finite difference method of TI-GQ with OpenMM (see Ref [6]). Only used when the `dec_int` is set to "TI". Default value is 0.001.

`itera_steps`: Number of simulation steps per iteration, with the total number of steps per window being this value multiplied by the number of iterations per window above.

`itcheck`: write OpenMM checkpoint file at every `itcheck` iterations.

`software`: Choose “openmm” for this option if using this simulation software.

6. Host configuration

6.1 Hosts provided with the GHOAT.py distribution

In the ./hosts-library folder, inside the GHOAT package, there are parameters and auxiliary files for all the hosts described the benchmark sets for binding free energies [15]. They are the compounds cucurbit[7]uril (CB7), α- and β-cyclodextrins (α-CD and β-CD), octa-acids (OA), and octa-acids with four methyl groups added to it (TEMOA). The parameters and initial coordinates for several ligands that bind to these systems can be found at the GitHub page of the benchmarks at https://github.com/MobleyLab/benchmarksets. That way, binding free energy calculations for all guest-host systems from this benchmark can be performed by GHOAT.py in an automated way.

In the benchmarks provided parameters, compounds CB7, OA and TEMOA are made up of a single residue and use GAFF atom types, so only the host anchors have to be defined for use with GHOAT. The α-CD and β-CD hosts are made up of multiple residues which have to be correctly connected, for which an auxiliary host_bonds file is needed. In addition, they use atom types and parameters from Ref. [16], so these also have to be included in the host configuration for the two CD species. More details on the setup of single and multiple residue hosts are explained in the next subsection, as well as in the README files from the host library.

6.2 Adding a new host

Hosts that are described by a single residue in their parameter and structure files only need their three anchors H1, H2 and H3 chosen for them to work with the GHOAT.py workflow. The main concern here is to avoid gimbal locks when restraints are applied, for example if the H3-H2-H1 or H2-H1-G1 angles approach 0^{o} or 180^{o}. It is also desirable that the the chosen anchors provide some structural stability of the host when the conformational distance restraints are used for this molecule (section 2.1). A criteria that fulfills these two requirements is to choose anchors that form a near equilateral triangle normal to the host symmetry axis. This is shown for the cucurbit[7]uril (CB7) host in Figure 1 (left) and Figure 2 (right), in which atoms named C15, C28 and C40 were picked from the structure provided by the guest-host benchmarks.

In the case of hosts that have multiple identical residues connected, an additional file for the host bonds between residues has to be added to the ./GHOAT/parameters/ folder. This is because GHOAT requires the host .mol2 parameter file to have only a single residue in it, either the whole host or one of its identical fragments. This new file is named host_bonds, and contains the information to add the necessary bonds using the Antechamber’s *leap* [1] format. For example, for additional bonds between the six fragments making up the α-CD host, the host_bonds file would have the following lines:

```
bond model.1.O1 model.2.C4
bond model.2.O1 model.3.C4
bond model.3.O1 model.4.C4
bond model.4.O1 model.5.C4
bond model.5.O1 model.6.C4
```

```
bond model.6.O1 model.1.C4
```

, which connect residues from one to 6 in sequence through their O1 and C4 atoms. The variable `final_host_num` also needs to be defined, in this case it is equal to “6”, the total number of residues from the host.

For hosts with single or multiple residues, it is possible to include parameters outside of GAFF or GAFF2. This is done by calling additional parameter files and defining new properties, such as atom types, in the beginning of the *leap* input file. The necessary commands have to be added to the end of the ./GHOAT/amber_files/tleap.in.amber16 file, which is the starting point for the building of all systems. Additional parameter files should be included in the ./GHOAT/parameters folder, and have to be in .dat format for GHOAT to recognize them. Examples for this procedure can be found for the α-CD and β-CD hosts included in the hosts library, with more information in their associated README files.

7. References

[1] D.A. Case, K. Belfon, I.Y. Ben-Shalom, S.R. Brozell, D.S. Cerutti, T.E. Cheatham, III, V.W.D. Cruzeiro, T.A. Darden, R.E. Duke, G. Giambasu, M.K. Gilson, H. Gohlke, A.W. Goetz, R. Harris, S. Izadi, S.A. Izmailov, K. Kasavajhala, A. Kovalenko, R. Krasny, T. Kurtzman, T.S. Lee, S. LeGrand, P. Li, C. Lin, J. Liu, T. Luchko, R. Luo, V. Man, K.M. Merz, Y. Miao, O. Mikhailovskii, G. Monard, H. Nguyen, A. Onufriev, F.Pan, S. Pantano, R. Qi, D.R. Roe, A. Roitberg, C. Sagui, S. Schott-Verdugo, J. Shen, C. Simmerling, N.R.Skrynnikov, J. Smith, J. Swails, R.C. Walker, J. Wang, L. Wilson, R.M. Wolf, X. Wu, Y. Xiong, Y. Xue, D.M. York and P.A. Kollman (2020), AMBER 2020, University of California, San Francisco.

[2] P. Eastman, J. Swails, J. D. Chodera, R. T. McGibbon, Y. Zhao, K. A. Beauchamp, L.-P. Wang, A. C. Simmonett, M. P. Harrigan, C. D. Stern, R. P. Wiewiora, B. R. Brooks, and V. S. Pande (2017). “OpenMM 7: Rapid development of high performance algorithms for molecular dynamics.” *PLOS Computational Biology,* **13**, e1005659.

[3] J. D. Chodera and M. R. Shirts (2011). "Replica exchange and expanded ensemble simulations as Gibbs multistate: Simple improvements for enhanced mixing." Journal of Chemical Physics, **135**, 194110.

[4] F. P. S. Souza, G. Heinzelmann and G. F. Caramori (2023) “Investigating the Solvent Effects on Binding Affinity of PAHs−ExBox4+ Complexes: An Alchemical Approach”. Journal of Physical Chemistry B, **127**, 249.

[5] G. Heinzelmann and M. K. Gilson (2021). “Automation of absolute protein-ligand binding free energy calculations for docking refinement and compound evaluation”. *Scientific Reports,* **11**, 1116.

[6] G. Heinzelmann, D. J. Huggins and M. K. Gilson (2024). “BAT2: an Open-Source Tool for Flexible, Automated, and Low Cost Absolute Binding Free Energy Calculations”. *Journal of Chemical Theory and Computation*, **20**, 6518.

[7] M. R. Shirts and J. Chodera (2008) “Statistically optimal analysis of samples from multiple

equilibrium states.” *Journal of Chemical Physics*, **129**, 129105.

[8] W. Humphrey, A. Dalke and K. Schulten. (1996) "VMD - Visual Molecular Dynamics", *Journal of Molecular Graphics*, **14**, 33-38.

[9] G. J. Rocklin, D. L. Mobley, K. A. Dill, and P. H. Hünenberger. (2013) “Calculating the binding free energies of charged species based on explicit-solvent simulations employing lattice-sum methods: An accurate correction scheme for electrostatic finite-size effects.” *Journal of Chemical Physics* **139**, 184103.

[10] J. Wang, W. Wang, P.A. Kollman, and D.A. Case.. (2006) "Automatic atom type and bond type perception in molecular mechanical calculations".*Journal of Molecular Graphics and Modelling*, **25**, 247-260.

[11] J. Wang, R.M. Wolf, J.W. Caldwell, and P. A. Kollman, D. A. Case (2004) "Development and testing of a general AMBER force field". *Journal of Computational Chemistry*, **25**, 1157-1174.

[12] A. Jakalian, B. L. Bush, D. B. Jack, and C.I. Bayly (2000) "Fast, efficient generation of high-quality atomic charges. AM1-BCC model: I. Method". *Journal of Computational Chemistry*, **21**, 132-146.

[13] I. S. Joung and T. E. Cheatham III (2008). “Determination of Alkali and Halide Monovalent Ion Parameters for Use in Explicitly Solvated Biomolecular Simulations”. *The Journal of Physical Chemistry B*, **112**, 9020-9041.

[14] C. W. Hopkins, S. Le Grand, R. C. Walker, and A. E. Roitberg. (2015) “Long-Time-Step Molecular Dynamics through Hydrogen Mass Repartitioning”. *Journal of Chemical Theory and Computation*, **11**, 1864-1874.

[15] D.L. Mobley, G. Heinzelmann, N.M. Henriksen, and M.K. Gilson. "Predicting binding free energies: Frontiers and benchmarks (a perpetual review)" - https://escholarship.org/uc/item/9p37m6bq.

[16] C. Cézard, X. Trivelli, F. Aubry, F. Djedaïni-Pilard and F.-Y. Dupradeau (2011). “Molecular dynamics studies of native and substituted cyclodextrins in different media: 1. Charge derivation and force field performances”. *Physical Chemistry Chemical Physics*, **13**, 15103-15121.