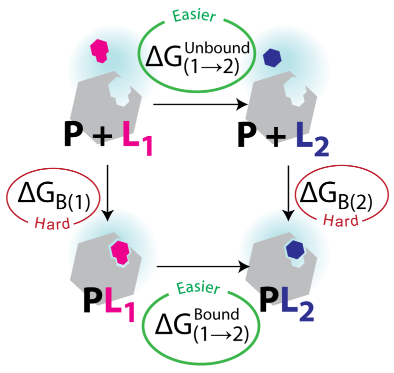
6. Free-energy perturbation for relative binding affinities

BSR3101: Computer Aided Drug Design

In this tutorial we will use FESetup and gromacs to perform a simple free-energy perturbation calculation of the relative binding affinity of three simple ligands binding to the lyzozyme. FESetup is a tool that streamlines the preparation of input for different molecular dynamics packages to perform free-energy calculations.
Relative binding free-energy calculations allow estimating the difference in binding affinity between ligands. The binding free-energy difference between two ligands is given by the difference

$$\Delta\Delta G_{AB} = \Delta G_B - \Delta G_A$$

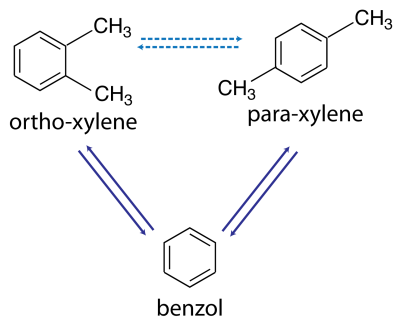
Where ΔG_B and ΔG_A are the absolute binding affinities of the two ligands. Since free-energy estimations have usually large error bars, the relative error for $\Delta\Delta G_{AB}$ can quickly become very large. By directly calculating the free-energy difference the accuracy of the estimate can be considerably improved, since common sources of error cancel out in the final result. Specifically, for two ligands binding to a given target, we consider the following thermodynamic cycle:


Easier
$\Delta G^{Unbound}_{(1\rightarrow 2)}$
P + L1
P + L2
$\Delta G_{B(1)}$
Hard
$\Delta G_{B(2)}$
Hard
PL1
PL2
Easier
$\Delta G^{Bound}_{(1\rightarrow 2)}$

We will setup a procedure to calculate the values of $\Delta G_{(1>2)}^{(Bound)}$ and $\Delta G_{(1>2)}^{(Unbound)}$, and then use these values to calculate

$$\Delta\Delta G_{AB} = \Delta G_{(B1)} - \Delta G_{(B2)} = \Delta G_{(1>2)}^{(Unbound)} - \Delta G_{(1>2)}^{(Bound)}$$

(since $\Delta G_{(B1)} + \Delta G_{(1>2)}^{(Bound)} - \Delta G_{(B2)} - \Delta G_{(1>2)}^{(Unbound)} = 0$). When several ligands are present, different perturbations can be used to calculate the relative free-energy differences of pairs of ligands, and the redundant values can be used to double-check the accuracy of the simulations. Specifically, calculating the free-energy difference along closed loops must give 0 (i.e. the free-energy difference between two ligands must be independent on the path used to transition between the two ligands).
In this exercise we will setup the following perturbations for binding at the lysozyme binding site:



1. Login to Minerva and download the files needed for this tutorial

We will setup a simple free-energy perturbation calculation on Minerva. Connect to Minerva by logging in to **`minerva2.hpc.mssm.edu`**. Make sure to set-up your connection to allow X11 forwarding to be able to display graphical interfaces of the programs on the remote machine you are using. (If you are running on a windows machine, also make sure that the X11 client xming is running). Once you are connected, change the working directory to the course scratch space, and create a new folder to save your work.
The cd command changes to directory to the folder just created. Keep in mind that UNIX commands are case sensitive (press Enter after each command to execute it).

```
> mkdir FEP_lyzozyme
> cd FEP_lyzozyme
```

(use your name, without spaces, instead of *yourname*)

Now, copy the tutorial folder, and extract it. This folder contains input files you'll need, as well as some output files that are needed to complete the analysis.

```
> cp /sc/orga/projects/BSR3101/tutorial_06.tar.gz ./
> tar xzf tutorial_06.tar.gz
```

This will create a folder **`tutorial_06`** containing the files we need for this exercise. Finally, before we begin, let us load some libraries that will be used to generate the simulation setup.

```
> module use /sc/orga/projects/BSR3101/softwares/modulefiles/
> module load fesetup
```

2. Generate the ligand and complex topologies

As discussed, a calculation of a relative free-energy requires several simulations.

- A first set of simulations converts ligand 1 to ligand 2 in solution, by running multiple MD trajectories with topologies that morph between the two final states.
- A second set of simulations converts ligand 1 to ligand 2 while bound to the protein (i.e. in a complex), again, by running multiple MD trajectories with topologies that morph between the two final states.

Before running the simulations we have to setup the necessary files. Specifically, given the ligands and the protein, we have to

- Parametrize the ligands, assigning the proper atom types and parameters for each interaction
- Create a topology that morphs between the two ligand topologies
- Parametrize the protein
- Create a simulation box that contains the ligand and solvent molecules
- Create a simulation box that contains the complex and solvent molecules

  While all these steps can be performed manually, we will use a library called FESetup that streamlines the preparation of the required files and can be customized to generate inputs for different MD engines (gromacs, NAMD, amber, etc.).

The relevant files we need to setup the simulations are in the input folder:

```
> cd tutorial_06/input
> ls
```

`ligands`	Folder that contains the structures of the ligands we want to simulate
`proteins`	Folder that contains the structures of the targets
`morph.in`	File that contains the information to create simulations of multiple ligands with multiple targets

Let's first have a look into the ligands folder. Change your working directory to this folder, and list its contents (press Enter after each command to execute it):

```
> cd ligands
> ls
```

You should see three folders named after the ligands whose information they contain

```
> cd benzol
> ls
> less ligand.mol2
```

```
@<TRIPOS>MOLECULE
181L.lig
   12    12     1     0     0
SMALL
bcc


@<TRIPOS>ATOM
      1 C1          25.9780    5.3270    4.7790 C.ar      1 LIG      -0.130000
      2 H1          25.2390    4.6940    5.2400 H         1 LIG       0.130000
      3 C2          26.3950    5.0740    3.4990 C.ar      1 LIG      -0.130000
      4 H2          25.9770    4.2470    2.9500 H         1 LIG       0.130000
      5 C3          27.3400    5.8600    2.9020 C.ar      1 LIG      -0.130000
      6 H3          27.6820    5.6300    1.9070 H         1 LIG       0.130000
      7 C4          27.8370    6.9210    3.5690 C.ar      1 LIG      -0.130000
      8 H4          28.5640    7.5610    3.0980 H         1 LIG       0.130000
      9 C5          27.4200    7.1960    4.8560 C.ar      1 LIG      -0.130000
     10 H5          27.8160    8.0500    5.3800 H         1 LIG       0.130000
     11 C6          26.4980    6.3790    5.4690 C.ar      1 LIG      -0.130000
     12 H6          26.1870    6.5640    6.4830 H         1 LIG       0.130000
@<TRIPOS>BOND
     1    2    1 1
     2    3    1 ar
     3    4    3 1
     4    5    3 ar
     5    6    5 1
     6    7    5 ar
     7    8    7 1
     8    9    7 ar
     9   10    9 1
    10   11    1 ar
    11   11    9 ar
    12   12   11 1
@<TRIPOS>SUBSTRUCTURE
     1 LIG          1 TEMP              0 ****  ****    0 ROOT
```

(press **q** to return to the command prompt)

We see the structure of the file contains information about the position of the atoms and the bonds connecting them. The first section specifies what is contained in the file (12 atoms, 12 bonds), that the molecule is a small-molecule, and that the charges are AM1-BCC (bond-charge correction) charges. The mol2 format is a standard format and ligand information from smiles or pdb can easily be converted to mol2 using the maestro interface.

Let us now look at the information required to prepare the ligands. We first go back to the **input** folder, and then inspect the **morph.in** file.

```
> cd ../../
> less morph.in
```

As customary in Unix systems, comment lines start with a pound sign (#). This file has several sections. It starts with a **global** section, where some general parameters of the simulation are set up. Specifically, we select the force-field we wish to use to describe our system and what MD engine we want to generate input files for.

```
# T4-lysozyme: setup for relative alchemical free energy simulations
# minimizing each ligand and creates morph mappings by minimizing
# the number of softcore atoms, create a complex with these morphs

logfile = T4-lysozyme_morph.log
forcefield = amber, ff14SB, tip3p, cm
#ff_addons = GLYCAM_06j-1, lipid14
AFE.type = gromacs
AFE.separate_vdw_elec = True

## USE this to minimize and equilibrate with amber
mdengine = amber, sander
#mdengine = amber, pmemd.MPI
#mdengine.prefix = mpirun -np 4

## USE this to minimize and equilibrate with gromacs
#mdengine = gromacs, mdrun
#mdengine.postfix = -nt 4
```

Then a **ligand** section follows, introduced by the line **`[ligand]`**. This section specifies which perturbations we want to simulate. In this case, we chose to run the following three simulations: benzol ➝ o-xylene, benzol ➝ p-xylene, o-xylene ➝ p-xylene. The choice of which simulations to run is an important step of the setup, as discussed, and can affect the efficacy of the simulation and accuracy of the results.
After specifying the name of the folder where the ligand information is located (**`ligands`**), and the convention used to name the ligand files (**`ligand.mol2`**), some parameters for the solvation of the ligands are defined.

```
[ligand]
basedir = ligands
file.name = ligand.mol2
morph_pairs = benzol > o-xylene,
        benzol > p-xylene,
        o-xylene > p-xylene

# the following are required to create the morph in solution
box.type = rectangular
box.length = 12.0
neutralize = yes
min.nsteps = 100
```

A **protein** section follows, starting with the keyword **`[protein]`**, specifying how to prepare the protein/DNA part of the simulation. Again, the name of the folder that contains the target(s) is indicated (**`proteins`**) and the name of the individual files that we want to use to setup the complexes with the ligands (in this case, we only use one target, **`181L`**). Protonation states are assigned using the Propka algorithm (Jensen Research Group , Department of Chemistry, Copenhagen).

```
# prepare the biomolecule (could also be DNA for example)
[protein]
basedir = proteins
molecules = 181L
propka = t
```

Finally, the input file is completed by a **complex** section, starting with **`[complex]`**. This section specifies the parameters for the creation of the complex topologies (size of solvation box, etc.). Also, a minimization of the structure can be requested at this point, specifying how many minimization steps are required.

```
# create the solvated complex and minimise it, from this create the
# solvated complex with the ligand morphs copied in
[complex]
# the following are required to create the morph in solution
box.type = rectangular
box.length = 12.0
align_axes = True
neutralize = yes

min.nsteps = 200
min.ncyc = 100
```

After making sure that the ligands and the proteins directories contain the required files, we can run the preparation step by typing
`> FESetup morph.in` (Enter)
The output should look like this:

```
=== FESetup release 1.2.1, SUI version: 0.8.3 ===

Please cite: HH Loeffler, J Michel, C Woods, J Chem Inf Mod, 55, 2485
             DOI 10.1021/acs.jcim.5b00368
For more information please visit http://www.ccpbiosim.ac.uk/fesetup/

Making biomolecule 181L...
Making ligand benzol...
Making ligand o-xylene...
Making ligand p-xylene...
Morphs will be generated for gromacs
Morphing benzol to o-xylene...
Morphing benzol to p-xylene...
Morphing o-xylene to p-xylene...
Making complex from 181L and benzol...
Making complex from 181L and p-xylene...
Making complex from 181L and o-xylene...
Creating complex 181L:benzol with ligand morph benzol~o-xylene...
Creating complex 181L:benzol with ligand morph benzol~p-xylene...
Creating complex 181L:o-xylene with ligand morph o-xylene~p-xylene...

=== All molecules built successfully ===
```

The process created four new directories, **`_ligands`**, **`_proteins`**, **`_complexes`**, and **`_perturbations`**. The first three contain intermediate files for the preparation, as well as input files to run standard MD simulations of the ligands and the complexes. The prepared files for free-energy perturbations are contained in the **`_perturbations`** folder, whose content we will now briefly examine.

Inside the **`_perturbations`** folder a **`gromacs`** folder has been created, since we requested input files in the format that the Gromacs MD code needs to run simulations. Let us first change our working directory to this folder, and list its content
`> cd _perturbations/gromacs/` (Enter)
`> ls` (Enter)

```
benzol~o-xylene
benzol~p-xylene
```

```
o-xylene~p-xylene
```

For each of the required perturbations, a folder named ligand1~ligand2 was created. Each of these folders contains the files needed to run simulations for the "solvent" leg and the "complex" leg of the thermodynamic cycle.

```
> cd benzol~o-xylene
> ls
complex              state0.atp              MORPH1.mol2
MORPH0.frcmod        state0.pdb              pert.itp
morph.gro            state1.vac.mdp          state0.itp
solvated             mcs_map.pkl             state1.atp
state0.parm7         MORPH1.frcmod           state1.pdb
state1.gro           pert.atp
state1.rst7          state0.gro
leap.log             state0.rst7
MORPH0.mol2          state1.parm7
morph.top            mcs.mol2
```

Let us inspect the files for the "solvent" simulations. The relevant ones are the **`morph.gro`** that contains the coordinates, as well as the **`morph.top`** and **`pert.atp`** files that contain the topology and parameters for the perturbation. The **`mdp`** file contains parameters for the simulation settings, which we will discuss in the next section.

```
> cd solvated
leap.log             morph.gro
morph.top            pert.atp
pert.itp             sol.mdp
state0.parm7         state0.pdb
state0.rst7          state1.parm7
state1.pdb           state1.rst7
ligand_removed.pdb
```

3. Setup the alchemic strategy

Once the topologies for the ligands and the complexes have been prepared, we have to decide two important details of the protocol:

- the equilibration strategy for the simulations
- the alchemic path connecting the two ligands and setup simulations for each intermediate conformation.

Equilibration strategy

A typical equilibration scheme consists of several simulation steps with different settings. The goal of such scheme is to produce a starting point for the production step that is as close as possible to a conformation that reflects the equilibrium of the system, therefore achieving optimal efficiency for the sampling of the terms needed for the final estimates of the free-energy.

The scheme used in the simulations was the following:

`em_steep.mdp`	Energy minimization with a steepest gradient algorithm. This removes any clash that may be present after the system setup
`nvt.mdp`	Short equilibration (100 ps) at constant volume
`npt.mdp`	Short equilibration (100 ps) at constant pressure (Parrinello-Rahman)
`prod.mdp`	Production run (10 ns) at constant pressure (Parrinello-Rahman)

This scheme was used to run the simulations for this tutorial, and the **`mdp`** files to implement this scheme are provided in the output. For the sake of simplicity, however, we will proceed with a single production step, without the equilibration.

```
; TI/FEP mdp template for solution
; Note: this is for Gromacs 2016 and later
integrator                = sd
ld-seed                   = -1
bd-fric                   = 0
dt                        = 0.001
nsteps                    = 500000
nstcomm                   = 100
```

Alchemical path

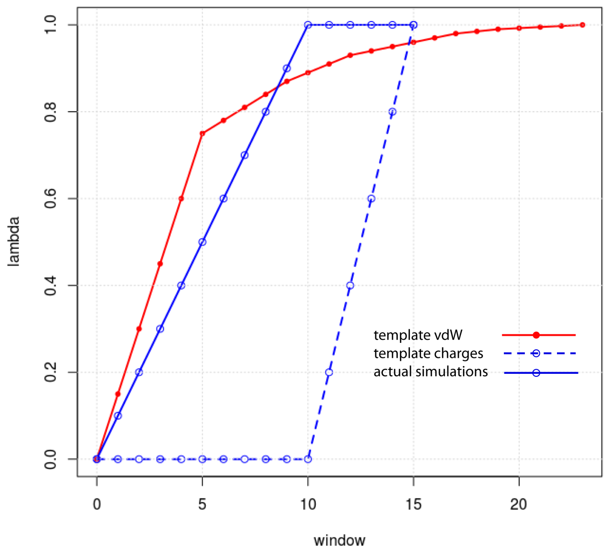
The second, important decision is the alchemic path to convert one ligand into another.
The **FESetup** library generates a template **mdp** file that can be customized to the chosen path. The section of the **mdp** file we are interested in starts with

```
; TI/FEP parameters
free-energy               = yes
delta-lambda              = 0
init-lambda-state         = %L%
; lambda paths: appearing atoms: vdW before q_on
fep-lambdas=  0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.2 0.4 0.6 0.8 1.0
vdw-lambdas=  0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.0 1.0 1.0 1.0 1.0
mass-lambdas= 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

nstdhdl                   = 100
calc-lambda-neighbors     = -1
```

This syntax specifies that we will run 17 "windows", or intermediate alchemical states. The first one, characherized by **fep-lambdas=0.0** and **vdw-lambdas=0.0**, corresponds to the topology of ligand 1. The second one starts to convert the vdw parameters of the ligand to the ones of ligand 2, but maintains the charges of the atoms to the ones of ligand 1 (**fep-lambdas=0.0** and **vdw-lambdas=0.1**). All intermediate states are defined accordingly, until the final state, that corresponds to ligand 2: **fep-lambdas=1.0** and **vdw-lambdas=1.0**.
In the template file, the additional atoms for o-xylene are created by switching on the vdW interactions first, and only subsequently switching on the electrostatic interactions.

In the actual production runs more windows were run with an increased window density at the end of the transformation (close to lambda=1).



The next section turns of the soft-core potentials to prevent erratic behavior of the energy estimates due to the divergent behavior of the non-bonded potentials, and defines some output details.

```
; soft core potential
sc-alpha                 = 0.5
sc-coul                  = no
sc-power                 = 1
sc-r-power               = 6
sc-sigma                 = 0.3

; output details
dhdl-derivatives         = yes
dhdl-print-energy        = no
separate-dhdl-file       = yes
dh_hist_size             = 0
dh_hist_spacing          = 0.1
```

For the purposes of this tutorial, we will skip the actual running of the simulations, described in section 4 below.

Proceed to the analysis of the pre-run output in section 5.

Warning: this step would take several hours on a single processor workstation. For the purposes of this tutorial, we will skip the actual execution of the calculation, and load instead the pre-calculated results.

4. Generate the run folders and run the simulations

For each of these the lambda states, we must create a separate folder to run the simulation of the corresponding window. Each of the simulation folders will have the same input files, except the parameter **`init-lambda-state`** that will be set to 0 for the first window, 1, for the second, and so on. The preparation can be achieved using a simple shell script (see below). A simple python code is also provided in the output files.

```
> mkdir MD
> for i in `seq 1 17`;
> do
> mkdir MD/State_$i
> sed "s/%L%/$i/g" _sol.mdp > MD/State_$1/sol.mdp
> done
```

After executing these commands, the **`MD`** folder now contains 17 folders ready to run the 17 windows for the selected perturbation. They can be run with gromacs by calling the following commands inside each folder

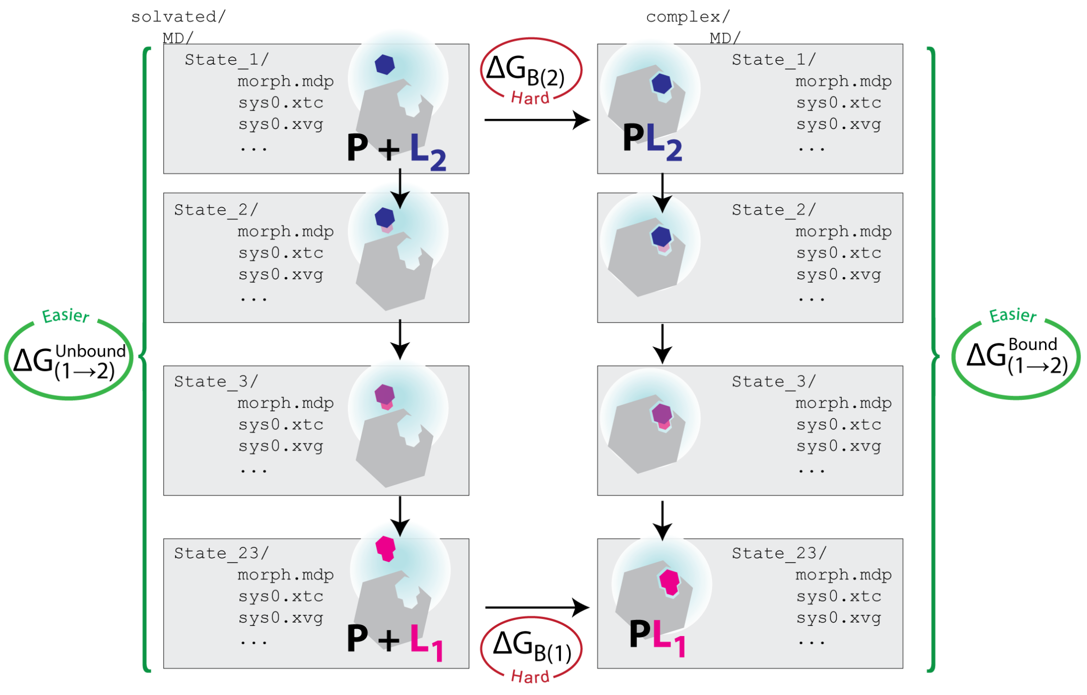
```
> module load gromacs/2016.3
> gmx_s grompp -f sol.mdp -p ../morph.top -c ../morph.gro -o ./sys0.tpr
> gmx_s mdrun -defnm sys0
```

In the output folders, a more realistic execution strategy is implemented.
In particular, the following details are changed:

1. the full equilibration protocol described above (minimization, NPT equilibration, NVT equilibration, production run) are implemented. For this a different mdp file is provided for each step and the appropriate parameters are changed to reflect the detail of each simulation step.

2. A different path, including more windows (24 and 34, respectively, for the solvated and complex legs of the thermodynamic cycle), is used.
3. The simulations are run on the gpu nodes by submitting a job to the queuing system rather than executing them on the front-end node.

After running the jobs, each folder contains the trajectory, as well as a file with the values of the energies and its derivatives that are needed for the calculation of the free-energy.



5. Analyze the results

The final step of the free-energy perturbation protocol is to calculate the free-energy differences ΔG_B and ΔG_A from the data in /solvated and /complex, respectively.
This can be achieved by different methods, and it's good practice to compare the differences to evaluate convergence and overlap problems between the windows.

All the required information is contained in the files sys0.xvg that contain the values of the energy H and the energy derivative dH/dl for different values of lambda. Then the change in free-energy for each step can be calculated using the different approaches and the total ΔGs obtained by summing along the solvated and the complex legs. The final free-energy difference is then calculated by taking the difference between the two partial values.

```
> cd output_small/benzol_o_xylene/solvated/
> module purge
> module load miniconda
> alchemical_analysis.py -d ./data/ -g -c -o . -u kcal -t 300. -f 10
```

The parameters have the following meanings:

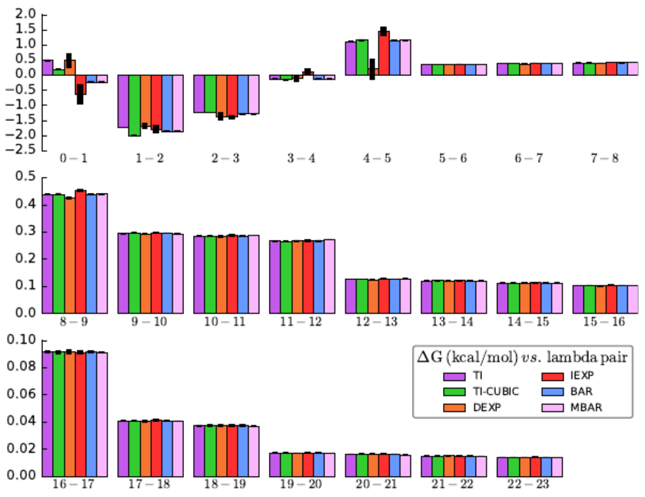
`-d ./data/`	Directory in which data files are stored. Default: Current directory.
`-g`	Plot the free energy differences evaluated for each pair of adjacent states for all methods, including the dH/dlambda curve for TI. Default: False.
`-c`	The Curve-Fitting-Method-based consistency inspector. Default: False.
`-o .`	Directory in which the output files produced by this script will be stored. Default: Same as datafile_directory.
`-u kcal`	Units to report energies: 'kJ', 'kcal', and 'kBT'. Default: 'kJ'
`-t 300.`	Temperature in K. Default: 298 K.
`-f 10`	Plot the free energy change as a function of time in both directions, with the specified number of points in the time plot. The number of time points (an integer) must be provided. Default: 0
`-s`	Discard data prior to this specified time as 'equilibration' data. Units picoseconds. Default: 0 ps.

The script has to be run on both the solvated dataset and the complex dataset, to obtain estimate of the $\Delta G_{(1>2)}^{(Unbound)}$ and $\Delta G_{(1>2)}^{(Bound)}$ values. Let us inspect the results of the analysis on the solvated perturbation (ligand in solvent). The results consist in files containing the free-energy differences, as well as some pdf files that allow assessing convergence problems.

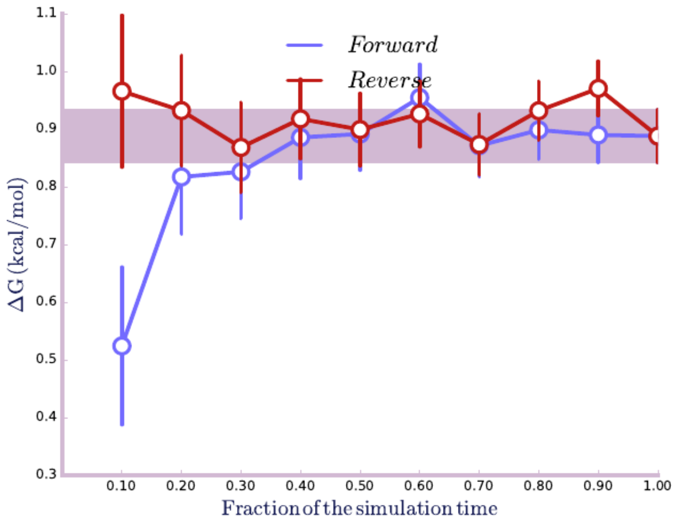
`> vi results.txt`

```
------------ --------------------- --------------------- --------------------
   States           TI (kcal/mol)       DEXP (kcal/mol)      MBAR (kcal/mol)
------------ --------------------- --------------------- --------------------
   0 -- 1         0.501  +-  0.017      0.499  +-  0.231    -0.223  +-  0.033
   1 -- 2        -1.734  +-  0.011     -1.672  +-  0.091    -1.848  +-  0.015
   2 -- 3        -1.234  +-  0.009     -1.356  +-  0.132    -1.276  +-  0.013
   3 -- 4        -0.120  +-  0.009     -0.089  +-  0.103    -0.123  +-  0.012
   4 -- 5         1.127  +-  0.011      0.213  +-  0.331     1.165  +-  0.014
   5 -- 6         0.362  +-  0.003      0.353  +-  0.004     0.370  +-  0.002
   6 -- 7         0.394  +-  0.003      0.385  +-  0.005     0.401  +-  0.002
   7 -- 8         0.420  +-  0.003      0.409  +-  0.005     0.426  +-  0.002
   8 -- 9         0.439  +-  0.003      0.425  +-  0.004     0.441  +-  0.002
   9 -- 10        0.295  +-  0.002      0.293  +-  0.003     0.295  +-  0.001
  10 -- 11        0.286  +-  0.002      0.285  +-  0.003     0.288  +-  0.001
  11 -- 12        0.267  +-  0.002      0.268  +-  0.003     0.273  +-  0.001
  12 -- 13        0.127  +-  0.001      0.124  +-  0.001     0.129  +-  0.000
  13 -- 14        0.122  +-  0.001      0.121  +-  0.001     0.122  +-  0.000
  14 -- 15        0.113  +-  0.001      0.113  +-  0.001     0.113  +-  0.000
  15 -- 16        0.104  +-  0.001      0.102  +-  0.001     0.103  +-  0.000
  16 -- 17        0.092  +-  0.001      0.092  +-  0.001     0.092  +-  0.000
  17 -- 18        0.041  +-  0.000      0.041  +-  0.001     0.041  +-  0.000
  18 -- 19        0.037  +-  0.000      0.038  +-  0.001     0.037  +-  0.000
  19 -- 20        0.017  +-  0.000      0.017  +-  0.000     0.017  +-  0.000
  20 -- 21        0.016  +-  0.000      0.016  +-  0.000     0.016  +-  0.000
  21 -- 22        0.015  +-  0.000      0.015  +-  0.000     0.015  +-  0.000
  22 -- 23        0.014  +-  0.000      0.014  +-  0.000     0.014  +-  0.000
------------ --------------------- --------------------- --------------------
  Coulomb:        1.701  +-  0.012      0.709  +-  0.446     0.888  +-  0.043
  vdWaals:        0.000  +-  0.032      0.000  +-  0.000     0.000  +-  0.000
    TOTAL:        1.701  +-  0.034      0.709  +-  0.446     0.888  +-  0.043
```

(note: estimates with other methods — TI-cubic, IEXP, BAR — are also included in the file and are not reported here for clarity). The same information is depicted graphically in the **`dF_state.pdf`** (which can be displayed with **`xpdf dF_state.pdf`**).



Furthermore, the comparison between the forward and backwards estimates (reported in the plots in the **`dF_t.pdf`** file) can be used to inspect convergence of the simulations.



6. Comparison with experimental values

The experimental binding free-energy difference between benzene and p-xylene is -0.52 ± 0.09 kcal/mol. Using the results obtained with the present simulations, we obtain for the same value an MBAR estimate in the complex of -1.634 ± 0.031 kcal/mol (see file), and for the solvated system -1.902 ± 0.039, which leads to the prediction

$$\Delta\Delta G = \Delta G_{(benz>p\text{-}xyl)}^{(Unbound)} - \Delta G_{(benz>p\text{-}xyl)}^{(Bound)} = -1.902 + 1.634 = -0.268 \pm 0.05$$

This estimate can be improved by observing that the complex simulation is still experiencing some convergence problems. Inspection of the time behavior of the estimates for the complex show that only in the last 10% of the simulation the forward and backward estimates are within each other's errorbars. We can restrict the estimate to this region using the –s flag:

```
> alchemical_analysis.py -d ./data/ -g -c -o . -u kcal -t 300. –s 9000
```

the new MBAR estimate for the complex simulation is now:

$$\Delta\Delta G = \Delta G_{(benz>p\text{-}xyl)}^{(Unbound)} - \Delta G_{(benz>p\text{-}xyl)}^{(Bound)} = -1.902 \ +1.517 \ = -0.385 \pm 0.06$$