Aim

Demonstrate how to combine Martini and CHARMM force fields using the virtual site (VS) scheme to simulate dual resolution membranes using capabilities of the Gromacs code.

Background

All-atomistic (AA) and coarse-grain (CG) simulations have been successfully applied to investigate a broad range of biomolecular processes. However, the accessible time and length scales of AA simulation are limited and the specific molecular details of CG simulation are simplified. Here, we propose a VS-based hybrid scheme [1] that can concurrently couple AA and CG resolutions in a single membrane simulation, mitigating the shortcomings of either representation. We demonstrate this method by combining the Martini 2.x CG force field [2] with the atomistic CHARMM36 [3] force fields and apply the method to lipid membrane systems [4].

In this tutorial, we shall start with combining CHARMM/Martini force fields for a simple alkane system, so you can appreciate the VS method, before jumping to a more complicated application like membranes. Before starting the journey in this tutorial, we highly recommend you to read another beautifully written tutorial about VS hybrid model combining OPLS/Martini force field (http://cgmartini.nl/index.php/tutorials-general-introduction-gmx5/tutorial-hybrid-model-using-virtual-sites-gmx5).

Setups and files for this tutorial can be found by downloading and unpacking (unzipping) the zip-file VShybridTutorial.zip (hyperlink here please). References to folders are made with respect to the parent folder that results after unpacking the zip file.

VS hybrid alkane

In this section, you need to build a box of dodecane to familiarize yourself with the hybrid model. This approach requires a mapping of the particles in the AA model to the CG beads, defining VS located at the center of mass of the corresponding AA atoms.

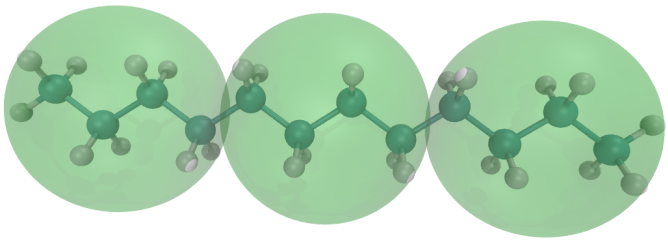

Figure 1. Illustration of VS hybrid dodecane. The bond and ball structure represents the AA model and the green transparent balls represent the VS beads. In the VS hybrid model, extra interaction sites defined by the center of the groups of atoms are added. (In Figure 1, the location of the sites are not shown, but they are at the center of the green transparent balls.) These interaction sites are defined as so-called virtual sites within Gromacs, and represent the CG model as a map of the AA model. By using this construction, the positions and dynamics of the AA and CG representations become closely linked.

We start the VS hybrid system from the AA CHARMM model. The CHARMM topology and input parameter files were produced by Ligand Reader & Modeler in Charmm-gui (https://www.charmm-gui.org/?doc=input/ligandrm). You can try this yourself (please refer to the Charmm-gui tutorials and examples; there is no description of the details in this tutorial), but if you want to focus on the VS hybrid building and simulation process, you can also find the files in the charmm_gui

folder, which includes the CHARMM36 topology (LIG.itp and topol.top) and coordinate (? and cgbuilder.gro) files for a single dodecane molecule.

Build the hybrid topology file

It is recommended to collect all the (modified) files for a VS hybrid run in a separate directory. The folder dodecane_worked contains the fully worked simulation. In the description below, we will use the name VSHRun as the name for the folder that has the simulation. You can then compare your results to those in dodecane_worked.

Starting from the CHARMM dodecane topology file (charmm_gui/LIG.itp), you just need to add the VS information to the topology file. Copy this file to VSHRun/LIG.itp and modify it as explained in this section.

In the Martini model, the dodecane is represented by three C1 beads shown in Figure 1 as the green transparent balls). To add these sites as virtual sites, two changes must be made to the topology file. Virtual sites are particles that belong to the molecule. Thus, you only need to put the VS beads (vC1) definition in the [atoms] directive.

```
[ atoms ]
; nr   type    resnr   residu  atom    cgnr    charge  mass
    1     CG331     1     LIG     C1     1    -0.270   12.0110   ; qtot -0.270
    2     CG321     1     LIG     C2     2    -0.177   12.0110   ; qtot -0.447
    3     CG321     1     LIG     C3     3    -0.183   12.0110   ; qtot -0.630
……
   36      HGA3     1      LIG    H24     36     0.090    1.0080   ; qtot -0.180
   37      HGA3     1      LIG    H25     37     0.090    1.0080   ; qtot -0.090
   38      HGA3     1      LIG    H26     38     0.090    1.0080   ; qtot  0.000
; the following defines the VS atoms
   39      vC1      1      VS     VC1     38     0        0
   40      vC1      1      VS     VC2     38     0        0
   41      vC1      1      VS     VC3     38     0        0
```

The positions of virtual sites are generated from positions of the atoms. This is what AA to CG mapping is all about. The correspondence of the VS sites to the AA atoms is specified in the [virtual_sitesn] directory. The following contents define the correspondence between the AA atoms and VS beads are therefore also added to the LIG. itp file:

```
[ virtual_sitesn ]
39 2  1 2 3 4
40 2  5 6 7 8
41 2  9 10 11 12
```

You can find the detailed information about the possible ways to define virtual sites in the Gromacs manual, but briefly, the first line in the [virtual_sitesn] directive shown above states that atom 39 is the center of mass (option 2 from virtual site options) of atoms 1, 2, 3, and 4 (which can be seen to be the first four C-atoms of the AA model). The mapping tells Gromacs how to calculate VS positions and forces on them from the atom positions and forces, and also how to calculate forces on the atoms from forces on the VS.
These are all the changes that need to be made to the LIG.itp to construct a VS hybrid topology. There are no CG bonded potentials (bonds, angles, dihedrals between VS or AA and VS particles) needed for our hybrid scheme.

Defining the interactions

We already have the .itp file for single hybrid dodecane (LIG.itp), but the CHARMM36 force field files do not know the new atom types (vC1) for the virtual sites. These must be combined with the CHARMM36 atom definitions and interactions. Here, we have done this for you; a more extensive explanation of how to build these files yourself is included in the section on the VS hybrid bilayer simulation. The file itp_file/ffnonbondH.itp contains the definitions of the virtual atom types. This file must be included in the system topology file (topol.top, copy this one from the charmm_gui folder). If we know that the number of molecules in the simulation is going to be 128 dodecane, we can already define 128 dodecane for the whole system in the .top file. We have made a ffnonbondH.itp file for you, how to build it is introduced in the next section. The two modifications of the file topol.top are shown below.

```
#include "../itp_file/ffnonbondH.itp"

[ molecules ]
LIG              128
```

Note that the include statement for the ffnonbondH.itp file must be added BEFORE the include statement of the charmm36.itp file. The number of molecules is specified at the bottom of the topol.top file.
Copy the charmm36.itp file from the charmm_gui directory to the working directory. In the charmm36.itp file, delete or comment out the lines referring [defaults] directive, because this is already defined in the ffnonbondH.itp file and a repetition results in a Gromacs error (in the gmx grompp step).

```
;[ defaults ]
; nbfunc        comb-rule       gen-pairs       fudgeLJ fudgeQQ
;1      2       yes     1.0     1.0
```

Build coordinate file

Now we are ready to build a coordinate file of a system of dodecane molecules, in our case, we will make a system of 128 dodecane molecules.
The CHARMM .gro file of a single dodecane look like this:

```
Header
  38
   1LIG     C1    1   0.618  -0.437  -0.000
```

```
   1LIG     C2    2   0.565  -0.294  -0.000
   1LIG     C3    3   0.412  -0.289  -0.000
…...
   1LIG    H24   36  -0.429   0.491  -0.090
   1LIG    H25   37  -0.429   0.491   0.090
   1LIG    H26   38  -0.576   0.438  -0.000
  0.00000   0.00000   0.00000
```

Copy this file to the working directory and give it the name one_aa_lig.gro. The three VS beads must be added to the end of the gro file. The coordinates of the VS beads are the center of mass of their corresponding CHARMM atoms. The number of VS beads should be added to the total atoms numbers list in the second line of the gro file. The final gro file is rendered in VMD and shown in Figure 1:

```
Header
  41
   1LIG     C1    1   0.618  -0.437  -0.000
   1LIG     C2    2   0.565  -0.294  -0.000
   1LIG     C3    3   0.412  -0.289  -0.000
…...
   1LIG    H24   36  -0.429   0.491  -0.090
   1LIG    H25   37  -0.429   0.491   0.090
   1LIG    H26   38  -0.576   0.438  -0.000
   1VS     VC1  39  -0.335   0.292  -0.000
   1VS     VC2  40   0.076   0.000   0.000
   1VS     VC3  41   0.488  -0.292  -0.000
  0.00000   0.00000   0.00000
```

With the coordinate of a single hybrid dodecane, we can create a box containing 128 dodecane molecule on the command line:

```
gmx_mpi insert-molecules -ci one_aa_lig.gro -box 4 4 4 -nmol 128 -o dodecane_box.gro -try 1000
```

Equilibrate and run the system

We have built the coordinate and topology files for 128 dodecane, thus we are ready for the simulation. The following command lines do a minimization of the initial system, an equilibration, and a (short) run:

```
gmx_mpi grompp -f minimization.mdp -p topol.top -c dodecane_box.gro -o mini_run.tpr -maxwarn 4
gmx_mpi mdrun -deffnm mini_run -v
gmx_mpi grompp -f martini_eq_2.mdp -p topol.top -c mini_run.gro -o eq.tpr -maxwarn 4
gmx_mpi mdrun -deffnm eq -v
gmx_mpi grompp -f martini_md.mdp -p topol.top -c eq.gro -o run.tpr -maxwarn 4
gmx_mpi mdrun -deffnm run -v
```

In fact, you are fooled by me. The VS in this model will not interact with each other and this simulation is equivalent to just simulating dodecane using the CHARMM force field, with three extra particles for each molecule that do nothing except show where the CG mapped particles are. The reason for not letting the VS interact with each other is that interactions between all dodecane molecules are already considered in AA resolution. To avoid double-counting the interactions in CG (VS) resolution, the interactions between VS in the "ffnonbondH.itp" file are defined with Lennard-Jones (LJ) interaction parameters that are zero. However, you have learned how to build a VS-based Martini model. We will combine the hybrid components with AA and CG molecules in the dual resolution membrane model. In this set-up, part of the system is fully coarse-grained and another part of the system is described by hybrid molecules. The hybrid molecules interact with the fully CG molecules by CG interactions.

Dual resolution membrane

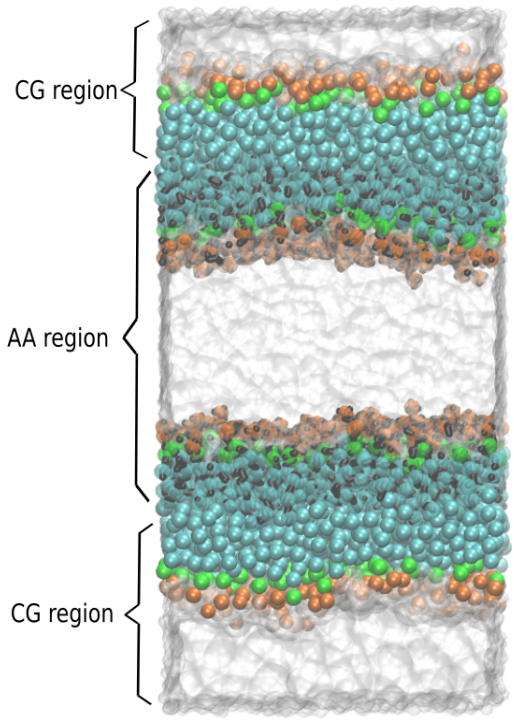

Figure 2. Virtual site hybrid membrane setups. Lipids in the AA region carry virtual sites. Lipids in the CG region are represented only by CG beads. Water in the AA region is atomistic (in this case TIP3P), water in the CG region is coarse-grain (in this case standard Martini water). Water particles are not shown; in both AA and CG regions, the water is shown as an enveloping surface. For both resolutions, lipid tail, linker, and head parts are represented by cyan, green, and orange color, respectively.

An intuitive way to build a dual resolution membrane system is a combination of AA membrane and CG water and put the resolution interface in the region around the lipid head of the membrane. However, it was shown that the VS hybrid AA/CG model fails to reproduce correct PMFs for the polar and charged AA solutes in CG solvent, yet properly reproduce the potentials of mean force (PMFs) between pairs of apolar amino acid side chain analogs [5]. Although possible, the set-up with hybrid

lipids and CG water results in severe artefacts (not published). Therefore, in this tutorial, the VS double-membrane systems are constructed as shown in Figure 2, to keep the resolution interface at the apolar lipid tail region. More details can be found in [4]. For the double membrane setup, any simulations requiring two membrane environments could benefit, e.g. to simulate the preliminary phase (stalk formation) of membrane fusion [6], or the effect of asymmetric ionic concentrations across the membrane [7], etc.

The full worked set-up for the dual resolution simulation is contained in the folder membrane_worked. It is advised that you create an empty folder yourself and work through the steps. You can compare your results to the one in the folder membrane_worked.

Build dual resolution AA and CG membrane system

Building a dual resolution membrane system can be done conveniently by using a number of tools developed for different purposes. In short, the procedure consists of four steps: (1) generating a coarse-grain membrane bilayer and water layer; (2) converting the CG lipids and water to AA VS hybrid lipids and AA water by backmapping; and (3) splitting the bilayer into leaflets, and shifting these to (4) combine original CG and backmapped parts to achieve a double membrane system, as shown in Figure 2. We use the insane code [8] to build the CG membrane system (1), from which the AA membrane (converted to hybrid lipids by adding VS as shown in the section on dodecane) and water are backmapped (2) using the backward software [9]. The original backward software can only backmap the CG membrane to the AA membrane. We tweak the library of the backward software to include the CG beads as VS in addition to the AA atoms, thus we build a hybrid membrane. You can find the modified library file here: backward/Mapping/dppc.charmm36.map. Note that python 2 is needed for backward and splitleafs softwares. The script splitleafs.py is downloaded from https://github.com/jbarnoud/splitleafs. The script is used for generating an index file that identifies upper and lower leaflets of the bilayer, which can be used to subsequently split the bilayer into leaflets (3) and combine them after shifting to create the double membrane set-up (4).

Create a new directory and run following command to build AA and CG membranes:

Step (1) is done by typing on the command line:

```
python3 ../insane/insane.py -o membrane_raw.gro -p top.top -x 8 -y 8 -z 10 -sol W -l DPPC=100
```

This creates a CG bilayer of pure DPPC with 100 lipids per leaflet in an 8x8x10 nm box and adds water beads.

Step (2) is done by typing on the command line:

```
python2.7 ../backward/backward.py -f membrane_raw.gro -o hybrid.gro -from martini -to charmm36 -sol
```

This creates a backmapped AA hybrid version of the CG membrane and water.

```
cp -r ../needed_file_charmm/* .
```

Build coordinate file

The following commands are used to build the hybrid membrane setup as illustrated in Figure 2. This setup is realized by shifting and combining the AA and CG membranes.

Step (3): identify lipids in upper and lower leaflets and create an index file with the atom indices of the two leaflets for the CG model:

```
python2.7 ../splitleaflet/splitleafs.py --atom DPPC:PO4 DIPC:PO4 CHOL:ROH W:W -r --membrane_raw.gro > tail_leaflets.ndx

echo "0"|gmx_mpi editconf -f membrane_raw.gro -n tail_leaflets.ndx -o lower_cg.gro
echo "1"|gmx_mpi editconf -f membrane_raw.gro -n tail_leaflets.ndx -o upper_cg.gro

gmx_mpi editconf -f upper_cg.gro -o upper_cg_moved.gro -translate 0 0 10

python2.7 ../splitleaflet/splitleafs.py --atom DPPC:VPO4 DIPC:VPO4 CHOL:VROH SOL:OW -r < hybrid.gro > tail_leaflets.ndx

echo "1"|gmx_mpi editconf -f hybrid.gro -n tail_leaflets.ndx -o upper_hb.gro
echo "0"|gmx_mpi editconf -f hybrid.gro -n tail_leaflets.ndx -o lower_hb.gro

gmx_mpi editconf -f lower_hb.gro -o lower_hb_moved.gro -translate 0 0 10
gmx_mpi editconf -f lower_hb_moved.gro -o lower_hb_moved.pdb
gmx_mpi editconf -f upper_hb.gro -o upper_hb.pdb

gmx_mpi editconf -f upper_cg_moved.gro -o upper_cg_moved.pdb
gmx_mpi editconf -f lower_cg.gro -o lower_cg.pdb

cat upper_hb.pdb lower_hb_moved.pdb upper_cg_moved.pdb lower_cg.pdb | grep -e "^ATOM" > whole.pdb

gmx_mpi editconf -f whole.pdb -resnr 1 -o whole.gro
```

Building the topology file

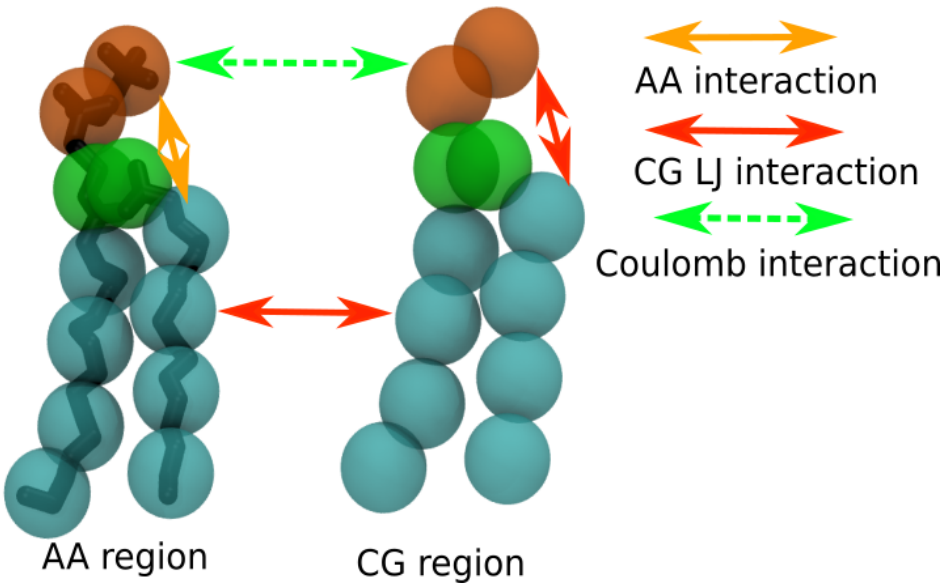

Figure 3. The multiscale system encompasses both AA/CG molecule representations. Each subsystem (AA and CG, respectively) interacts within itself according to its own resolution. Molecules in the AA region interact at the CG level with the molecules in the CG region through virtual sites that represent the AA region. When employing full electrostatics (PME), long-range electrostatic interactions between AA and CG charges are included (green dashed line). The AA region is shown as bonds, the virtual sites representing the AA system as well as the CG region are shown as spheres. Lipid tail, linker, and head are represented by cyan, green and orange color, respectively.

The interactions in the VS hybrid scheme used in this tutorial are schematically shown in Figure 3 for lipid molecules. The main idea of this VS hybrid scheme is that AA particles interact with AA particles according to the AA force field, and CG particles interact with CG particles according to the CG force field. AA particles do not interact directly with CG particles, but the interaction between molecules at AA resolution with molecules at CG resolution is achieved through VS. These VS carry interactions at CG level, i.e. the AA molecule 'sees' the embedding surroundings at the level of the CG model. Conversely, the CG molecules 'see' the AA molecules as CG molecules. To accomplish this interaction scheme, we have built a ffnonbondH.itp file for you in the itp_file folder, and which was already used in the dodecane set-up. The pair LJ interactions are defined in the [nonbond_params] directory. For Coulomb interactions, we use PME, thus all the AA/CG atoms carrying changes can interact with each other. Note that VS beads do not carry charges.

The following lines look complicated, but just to automatically build the topology file. You can copy these lines to your terminal or build a topology file by hand on your own.

```
tail -n +2 top.top > top2.top
cat ../needed_file_charmm/top_include.top top2.top>top.top
rm top2.top
```

```
python2.7 ../splitleaflet/splitleafs.py --atom SOL:OW -r -- hybrid.gro 2> sol_num_file.xvg > tail_leaflets.ndx
python2.7 ../splitleaflet/splitleafs.py --atom W:W -r -- membrane_raw.gro 2>w_num_file.xvg > tail_leaflets.ndx

cat sol_num_file.xvg|awk '{print $1 $2/3}'>solvent_num_file.xvg
rm sol_num_file.xvg


head -n -1 whole.gro > whole1.gro
tail -n 1  membrane_raw.gro |awk '{print $1" "$2" "$3*2}' >>whole1.gro
cp whole1.gro whole.gro

upper_leaflet_w=$(cat w_num_file.xvg |grep "upper_leaflet" | awk -F ':' '{print $2}'| awk '{print $1}')
lower_leaflet_w=$(cat w_num_file.xvg |grep "lower_leaflet" | awk -F ':' '{print $2}'| awk '{print $1}')
upper_leaflet_sol=$(cat solvent_num_file.xvg |grep "upper_leaflet" | awk -F ':' '{print $2}')
lower_leaflet_sol=$(cat solvent_num_file.xvg |grep "lower_leaflet" | awk -F ':' '{print $2}')
dppc_num=$(cat top.top |grep -m 1 "^DPPC "| awk  '{print $2}')


cat << EOF > top.top
#include "ffnonbondH.itp"
#include "DPPC.itp"
#include "TIP3.itp"
#include "dppc_CG.itp"



[ system ]
; name
INSANE! Membrane UpperLeaflet>POPC=3.0 LowerLeaflet>POPC=3.0


[ molecules ]
; name  number
DPPC          100
TIP3          $upper_leaflet_sol
DPPC          100
TIP3          $lower_leaflet_sol
DPPCC         100
W             $upper_leaflet_w
DPPCC         100
W             $lower_leaflet_w
```

EOF

Equilibrate the hybrid system

gmx_mpi grompp -f ../needed_file_charmm/nothing.mdp -c whole.gro -o index.tpr -p top.top -maxwarn 10

echo -e "del 1-15\nr DPPC\nname 1 bilayer_hybrid_AA\nr VDPPC \nname 2 VS\nr DPPCC \nname 3 bilayer_pure_CG\nr W\nname 4 W\nr TIP3\nname 5 SOL\nr DPPCC W \nname 6 CG\nr DPPC TIP3\nname 7 AA\nq\n" | gmx_mpi make_ndx -f index.tpr -o index.ndx

gmx_mpi grompp -f minimization.mdp -c whole.gro -p top.top -o mini_run_1.tpr -maxwarn 1 -n index.ndx

gmx_mpi mdrun -deffnm mini_run_1 -v

gmx_mpi grompp -f martini_eq_2.mdp -c mini_run_1.gro -p top.top -o mini_run_2.tpr -maxwarn 3 -n index.ndx -r mini_run_1.gro

gmx_mpi mdrun -deffnm mini_run_2 -v

gmx_mpi grompp -f minimization.mdp -c mini_run_2 -p top.top -o mini_run_2_1.tpr -maxwarn 1 -n index.ndx

gmx_mpi mdrun -deffnm mini_run_2_1 -v

gmx_mpi grompp -f martini_eq_3.mdp -c mini_run_2_1.gro -p top.top -o mini_run_3.tpr -maxwarn 3 -n index.ndx -r mini_run_2.gro

gmx_mpi mdrun -deffnm mini_run_3 -v

gmx_mpi grompp -f martini_eq_4.mdp -c mini_run_3.gro -p top.top -o mini_run_4.tpr -maxwarn 3 -n index.ndx -r mini_run_3.gro

gmx_mpi mdrun -deffnm mini_run_4 -v

Run the hybrid system

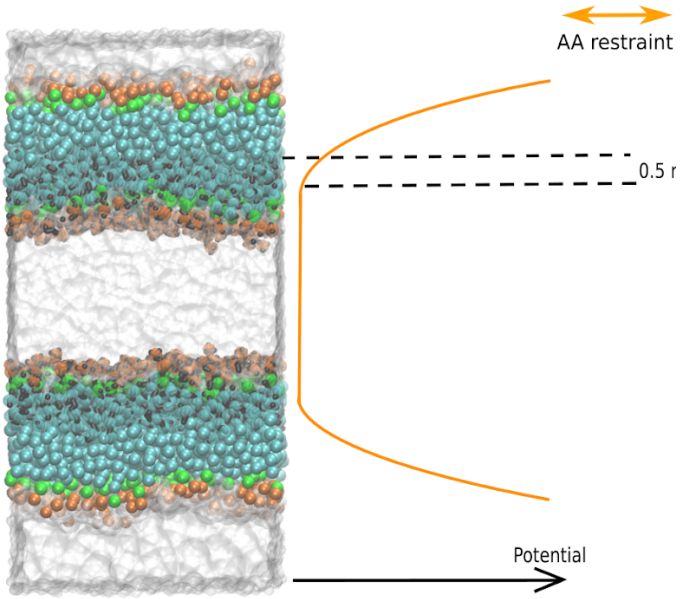

Figure 4. Illustration of flat bottom potential. The flat bottom potential is applied only on AA water molecules. The distance between the resolution interface and the beginning of flat bottom potential is 0.5 nm. The color scheme is the same as that of Figure 2.

The flat bottom potential is used to restrain the AA water and stop them from crossing the membrane, as shown in Figure 4. To accomplish this, we write a python code to project the coordinates of AA water on a xy plane, located at the center of the bulk AA water in the z-axis, and build the restrain gro file (test_FBP.gro) for you. Thus, if AA water molecules diffuse too far from the water plane in the z-axis will be pushed back.

```
python3 ../build_restrain_gro_file/flat_bottom_potential.py
gmx_mpi grompp -f martini_md.mdp -c mini_run_4.gro -p top.top -o run.tpr -maxwarn 3 -n index.ndx -r test_FBP.gro
gmx_mpi mdrun -deffnm  run -v
```

Conclusion

Congratulations! You have grasped how to combine Martini and CHARMM force fields using the virtual site(VS) to simulate dual resolution membranes. For now, this hybrid scheme only supports DPPC membranes and you are welcome to extend it to other membranes. The rule of thumb is that you can adjust the interactions between two resolutions while keeping the AA and CG potential unchanged. The aim is to reproduce the correct conformational or even dynamical properties, e.g. area per lipid, partial density, order parameter, diffusion constant, etc. against the AA and CG references.

References

1. Rzepiela AJ, Louhivuori M, Peter C, Marrink SJ. Hybrid simulations: combining atomistic and coarse-grained force fields using virtual sites. Phys Chem Chem Phys. 2011;13: 10437–10448.

2. Marrink SJ, Jelger Risselada H, Yefimov S, Tieleman DP, de Vries AH. The MARTINI Force Field: Coarse Grained Model for Biomolecular Simulations. J Phys Chem B. 2007;111: 7812–7824.

3. Vanommeslaeghe K, Hatcher E, Acharya C, Kundu S, Zhong S, Shim J, et al. CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields. J Comput Chem. 2009. p. NA–NA. doi:10.1002/jcc.21367

4. Liu Y, De Vries AH, Barnoud J, Pezeshkian W, Melcr J, Marrink SJ. Dual Resolution Membrane Simulations Using Virtual Sites. J Phys Chem B. 2020;124: 3944–3953.

5. Wassenaar TA, Ingólfsson HI, Prieß M, Marrink SJ, Schäfer LV. Mixing MARTINI: Electrostatic Coupling in Hybrid Atomistic–Coarse-Grained Biomolecular Simulations. J Phys Chem B. 2013;117: 3516–3530.

6. Markvoort AJ, Marrink SJ. Lipid acrobatics in the membrane fusion arena. Curr Top Membr. 2011;68: 259–294.

7. Khalili-Araghi F, Ziervogel B, Gumbart JC, Roux B. Molecular dynamics simulations of membrane proteins under asymmetric ionic concentrations. J Gen Physiol. 2013;142: 465–475.

8. Wassenaar TA, Ingólfsson HI, Böckmann RA, Tieleman DP, Marrink SJ. Computational Lipidomics with insane: A Versatile Tool for Generating Custom Membranes for Molecular Simulations. J Chem Theory Comput. 2015;11: 2144–2155.

9. Wassenaar TA, Pluhackova K, Böckmann RA, Marrink SJ, Tieleman DP. Going Backward: A Flexible Geometric Approach to Reverse Transformation from Coarse Grained to Atomistic Models. J Chem Theory Comput. 2014;10: 676–690.