Tutorial 5. SIRAH force field in GROMACS

Simulation of coarse grained lipid bilayers in explicit solvent

By Exequiel Barrera
Mail any comment or suggestion to *spantano@pasteur.edu.uy*

This tutorial shows how to use the SIRAH force field to perform a coarse-grained (CG) simulation of a DMPC bilayer in explicit solvent (called WatFour, WT4) in four simple steps: 1 download; 2 map; 3 solvate and; 4 run. The main references for this tutorial are: Barrera et al. *SIRAH Lipids* [JCTC, **2019**, 15:5674], Machado et al. *SIRAH Tools* [Bioinformatics, **2017**, 32:1568]. We strongly advise you to read these articles before starting the tutorial.

Required Software
GROMACS 4.6.7 or later version properly installed in your computer. The molecular visualization program VMD (freely available at *www.ks.uiuc.edu/Research/vmd*). The plotting software Grace (*plasma-gate.weizmann.ac.il/Grace*).

Prior knowledge
How to perform a standard atomistic molecular dynamic simulation with GROMACS.

Hands on
0) Download the file *sirah_[version].gmx.tgz* from *www.sirahff.com* and uncompress it into your working directory. Notice: *[version]* should be replaced with the actual package version e.g.: x2_18-09

```
tar -xzvf sirah_[version].gmx.tgz
```

You will get a folder *sirah_[version].ff/* containing the force field definition, the SIRAH Tools in *sirah_[version].ff/tools/*, molecular structures to build up systems in *sirah_[version].ff/PDB/*, frequently asked questions in *sirah_[version].ff/tutorial/SIRAH_FAQs.pdf* and the required material to perform the tutorial in *sirah_[version].ff/tutorial/5/* .

Make a new folder for this tutorial in your working directory:

```
mkdir tutorial5; cd tutorial5
```

Create the following symbolic links in the folder *tutorial5*:

```
ln -s ../sirah_[version].ff sirah.ff
ln -s sirah.ff/residuetypes.dat
ln -s sirah.ff/specbond.dat
```

Notice: Files *residuetypes.dat* and *specbond.dat* are essential for the correct definition of molecular groups and auto-detection of disulfide bonds and cyclic DNA polymers.

1) Map the atomistic structure of the preassembled DMPC bilayer to its CG representation:

```
./sirah.ff/tools/CGCONV/cgconv.pl\
 -i sirah.ff/tutorial/5/DMPC64.pdb\
 -o DMPC64_cg.pdb\
 -a sirah.ff/tools/CGCONV/maps/tieleman_lipid.map
```

Notice: By default no mapping is applied to atomistic lipids, as there is no standard naming convention for them. So users are requested to append a MAP file from the list in Table 1, by setting the flag *-a* in *cgconv.pl*. We recommend using PACKMOL (free at *http://m3g.iqm.unicamp.br/packmol*) for building the system. Reference building-block structures are provided at folder *sirah.ff/PDB/*, which agree with the mapping scheme in *sirah.ff/tools/CGCONV/maps/tieleman_lipid.map*. The provided DMPC bilayer contains 64 lipid molecules per leaflet distributed in a 6.4 * 6.4 nm surface, taking into account an approximate area per lipid of 0.64 nm^2 at 333 K. The starting configuration was created with the input file *sirah.ff/tutorial/5/DMPC_bilayer.pkm.* See *SIRAH_FAQs.pdf* for cautions on mapping lipids to SIRAH and tips on using fragment-based topologies.

Notice: This is an advanced usage of the script *cgconv.pl*, you can learn more from its help:

```
./sirah.ff/tools/CGCONV/cgconv.pl -h
```

The input file *DMPC64.pdb* contains all the heavy atoms composing the lipids, while the output *DMPC64_cg.pdb* preserves a few of them. Please check both PDB structures using VMD:

```
vmd -m sirah.ff/tutorial/5/DMPC64.pdb DMPC64_cg.pdb
```

From now on it is just normal GROMACS stuff!

Notice: GROMACS' commands in 5.x and later versions start with "*gmx*".

2) Use pdb2gmx to convert your PDB file into GROMACS format:

```
pdb2gmx -f DMPC64_cg.pdb -o DMPC64_cg.gro
```

When prompted, choose "*SIRAH force field"* and then "*SIRAH solvent models"*.

Notice: Getting warning messages of long bonds is fine and expected due to the CG nature of the residue topologies. However missing atom messages are errors which probably trace back to the mapping step. In that case, check your atomistic and mapped structures and do not carry on the simulation until the problem is solved.

3) Solvate the system

Define the simulation box of the system:

```
editconf -f DMPC64_cg.gro -o DMPC64_cg_box.gro -box 6.6 6.6 10 -c
```

Notice: As PACKMOL does not consider periodicity while building up the system, increasing the box size a few Angstroms may be required to avoid bad contacts between images.

Add WT4 molecules:

```
genbox -cp DMPC64_cg_box.gro -cs sirah.ff/wt416.gro -o DMPC64_cg_sol1.gro
```

Notice: in GROMACS 5.x and later versions the command genbox was renamed to "*gmx solvate"*

Edit the *[molecules]* section in *topol.top* to include the number of added WT4 molecules:

Hint! If you forget to read the number of added WT4 molecules from the output of genbox, then use the following command line to get it

```
grep -c WP1 DMPC64_cg_sol1.gro
```

Topology before editing	Topology after editing

`[ molecules ]`
`; Compound #mols`
`Lipid_chain_A 1`
`Lipid_chain_B 1`
	`[ molecules ]`
	`; Compound #mols`
	`Lipid_chain_A 1`
	`Lipid_chain_B 1`
	`WT4 850`

Notice: The number of added WT4 molecules (850) may change according to the software version.

Remove misplaced WT4 molecules inside the bilayer (read *sirah.ff/0ISSUES* and *SIRAH_FAQs.pdf* for comments on known solvation problems):

```
grompp\
 -f sirah.ff/tutorial/5/CPU/em_CGLIP.mdp\
 -p topol.top\
 -c DMPC64_cg_sol1.gro\
 -o DMPC64_cg_sol1.tpr

echo -e "a BE* BC1* BC2* BCT*\n name 7 tail\n\nq" |
make_ndx -f DMPC64_cg_sol1.gro -o DMPC64_cg_sol1.ndx
```

```
g_select\
 -f DMPC64_cg_sol1.gro\
 -s DMPC64_cg_sol1.tpr\
 -n DMPC64_cg_sol1.ndx\
 -on rm_close_wt4.ndx\
 -select "not (same residue as (resname WT4 and within 0.4 of group tail))"

editconf -f DMPC64_cg_sol1.gro -o DMPC64_cg_sol2.gro -n rm_close_wt4.ndx
```

Edit the *[molecules]* section in *topol.top* to correct the number of WT4 molecules:

Hint! Use the following command to get the remaining WT4 molecules in the system.

```
grep -c WP1 DMPC64_cg_sol2.gro
```

Add CG counterions and 0.15M NaCl:

```
grompp\
 -f sirah.ff/tutorial/5/CPU/em_CGLIP.mdp\
 -p topol.top\
 -c DMPC64_cg_sol2.gro\
 -o DMPC64_cg_sol2.tpr

genion\
 -s DMPC64_cg_sol2.tpr\
 -o DMPC64_cg_ion.gro\
 -np 21 -pname NaW\
 -nn 21 -nname ClW
```

When prompted, choose to substitute WT4 molecules by ions.

Notice: The available ionic species in SIRAH force field are: Na+ (NaW), K+ (KW) and Cl- (ClW). One ion pair (e.g. NaW-ClW) each 34 WT4 molecules renders a salt concentration of ~0.15M (see Appendix 1). If needed, we recommend adding counterions according to Machado et al. *SPLIT* [JCTC, **2020**].

Edit the *[molecules]* section in *topol.top* to include the CG ions and the correct number of WT4.

Before running the simulation it may be a good idea to visualize your molecular system. CG molecules are not recognized by molecular visualizers and will not display correctly. To fix this problem you may generate a PSF file of the system using the script *g_top2psf.pl*:

```
./sirah.ff/tools/g_top2psf.pl -i topol.top -o DMPC64_cg_ion.psf
```

Notice: This is the basic usage of the script *g_top2psf.pl*, you can learn other capabilities from its help:

```
./sirah.ff/tools/g_top2psf.pl -h
```

Use VMD to check how the CG system looks like:

```
vmd DMPC64_cg_ion.psf DMPC64_cg_ion.gro -e sirah.ff/tools/sirah_vmdtk.tcl
```

Notice: VMD assigns default radius to unknown atom types, the script *sirah_vmdtk.tcl* sets the right ones. It also provides a kit of useful selection macros, coloring methods, backmapping utility and a command to calculate and display the secondary structure of SIRAH proteins. Use the command *sirah_help* in the Tcl/Tk console of VMD to access the manual pages.

4) Run the simulation

The folder *sirah.ff/tutorial/5/CPU/* contains typical input files for energy minimization (*em_CGLIP.mdp*), equilibration (*eq_CGLIP.mdp*) and production (*md_CGLIP.mdp*) runs. Please check carefully the input flags therein.

Create an index file:

```
echo "q" | make_ndx -f DMPC64_cg_ion.gro -o DMPC64_cg_ion.ndx
```

Notice: WT4 and CG ions (NaW, ClW) are set automatically to the group "*SIRAH-Solvent*" while DMPC (named CMM at CG level) is assigned to group "*Lipid*".

Make a new folder for the run:

```
mkdir run; cd run
```

Energy Minimization:

```
grompp\
 -f  ../sirah.ff/tutorial/5/CPU/em_CGLIP.mdp\
 -p  ../topol.top\
 -po    em.mdp\
 -n  ../DMPC64_cg_ion.ndx\
 -c  ../DMPC64_cg_ion.gro\
 -o     DMPC64_cg_em.tpr

mdrun -deffnm DMPC64_cg_em &> EM.log &
```

Equilibration:

```
grompp\
 -f  ../sirah.ff/tutorial/5/CPU/eq_CGLIP.mdp\
 -p  ../topol.top\
 -po    eq.mdp\
 -n  ../DMPC64_cg_ion.ndx\
 -c     DMPC64_cg_em.gro\
 -r     DMPC64_cg_em.gro\
 -o     DMPC64_cg_eq.tpr

mdrun -deffnm DMPC64_cg_eq &> EQ.log &
```

Production (100ns):

```
grompp\
 -f  ../sirah.ff/tutorial/5/CPU/md_CGLIP.mdp\
 -p  ../topol.top\
 -po    md.mdp\
 -n  ../DMPC64_cg_ion.ndx\
 -c     DMPC64_cg_eq.gro\
 -o     DMPC64_cg_md.tpr

mdrun -deffnm DMPC64_cg_md &> MD.log &
```

Notice: You can find example input files for the GPU-CPU version of *mdrun* at folder *GPU/* within *sirah.ff/tutorial/5/* . GPU flags were set for GROMACS 4.6.7, other versions may complain about some specifications.

That's it! Now you can analyze the trajectory.

Example of trajectory analysis

Process the output trajectory at folder *run/* to account for the Periodic Boundary Conditions (PBC):

```
trjconv\
 -s DMPC64_cg_em.tpr\
 -f DMPC64_cg_md.xtc\
 -o DMPC64_cg_md_pbc.xtc\
 -n ../DMPC64_cg_ion.ndx\
 -pbc mol
```

When prompted, choose "*System*" for output.

Now you can check the simulation using VMD:

```
vmd ../DMPC64_cg_ion.psf ../DMPC64_cg_ion.gro DMPC64_cg_md_pbc.xtc\
 -e ../sirah.ff/tools/sirah_vmdtk.tcl
```

Calculate the area per lipid

```
g_energy -f DMPC64_cg_md.edr -o box_XY.xvg
```

When prompted, choose "Box-X" and "Box-Y" for output. End your selection with a zero

To calculate the area per lipid divide the membrane's area by the DMPC molecules per leaflet

$$Area/Lipid = \frac{Box(x) * Box(y)}{64}$$

Density profiles and bilayer thickness
Include a new group for phosphate beads in the index file:

```
echo -e "a BFO\nq\n" |
make_ndx -f DMPC64_cg_em.gro -n ../DMPC64_cg_ion.ndx -o DMPC64_cg_ion.ndx
```

```
g_density\
 -sl 1000\
 -ng 5\
 -f DMPC64_cg_md_pbc.xtc\
 -s DMPC64_cg_em.tpr\
 -n DMPC64_cg_ion.ndx\
 -o density_profile.xvg
```

When prompted, choose “Lipid”, “WT4”, “NaW”, “ClW” and “BFO”

Use Grace to plot the results:

```
xmgrace -nxy density_profile.xvg
```

The thickness of the bilayer is the distance between the two peaks corresponding to the position of phosphate beads (BFO) along the z-axis.

Mapping atomistic lipids to SIRAH

Table 1. Available mapping files (MAPs) at folder *sirah.ff/tools/CGCONV/maps/* for converting atomistic lipid structures to SIRAH models. Important! MAPs can not inter-convert different name conventions, e.g. *amber_lipid.map* won't generate fragment-based residues from residue-based force fields. Due to possible nomenclature conflicts, users are advised to check and modify the MAPs as required.

MAP	Type[1]	Compatibility	Source
amber_lipid.map	F	AMBER Lipid11-17 force fields	AMBER HTMD
GAFF_lipid.map	R	AMBER GAFF force field	LipidBook
charmm_lipid.map	R	CHARMM 27/36 force field, and "CHARMM compatible" GAFF nomenclature	CHARMM-GUI GROMACS LipidBook MemBuilder HTMD VMD
slipids.map	R	Stockholm lipids force field	SLIPIDS MemBuilder
OPLSA-AA_2014_lipid.map	R	All-atoms lipids for OPLS force field	Maciejewski et al. 2014
OPLSA-UA_lipid.map	R	United-atom lipids for OPLS force field	LipidBook
GROMOS43a1_lipid.map	R	United-atom lipids for GROMOS 43a1 and CKP force fields	LipidBook MemBuilder
GROMOS43a1-s3_lipid.map	R	United-atom lipids for GROMOS 43a1-s3 force field	GROMACS repo LipidBook MemBuilder
GROMOS53a6_lipid.map	R	United-atom lipids for GROMOS 53a6 force field	GROMACS repo LipidBook MemBuilder
tieleman_lipid.map	R	Berger lipids as implemented by Tieleman et al. for GROMOS force fields.	Tieleman LipidBook

[1] Fragment-based (F) or Residue-based (R) topology.

Appendix 1: Calculating ionic concentrations

$\rho_{WT4} = \rho_{H2O} = 1000$ g/L
$MW_{H2O} = 18$ g/mol
1 WT4 ~ 11 H_2O

$$M = \frac{mol}{V} \quad ; \quad n = mol\, N_A \quad ; \quad \rho = \frac{m}{V} \quad ; \quad m = mol\, MW$$

$$V = \frac{m}{\rho} = \frac{mol\, MW_{H_2O}}{\rho} = \frac{n_{H_2O} MW_{H_2O}}{N_A \rho} \quad ; \quad M = \frac{mol}{V} = \frac{n_{ion}}{N_A V} = \frac{n_{ion}}{N_A} \frac{N_A \rho}{n_{H_2O} MW_{H_2O}} = \frac{n_{ion} 1000}{n_{WT4}(11)(18)} \sim 5 \frac{n_{ion}}{n_{WT4}}$$

Number of WT4 molecules per ion at 0.15M: $n_{WT4} = 5 \frac{n_{ion}}{M} = \frac{5(1)}{0.15} \sim 34$