Tutorial 3. SIRAH force field in GROMACS

Simulation of coarse grained proteins in explicit solvent

By Matias Machado
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 protein 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: Darré et al. *WAT4?* [JCTC, **2010**, 6:3793], Machado et al. *SIRAH 2.0* [JCTC, **2019**, 15:2719], Machado et al. *SIRAH Tools* [Bioinformatics, **2017**, 32:1568]. We strongly advise you to read these articles before starting the tutorial.

Required Software
GROMACS 4.5.5 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*) and the R statistical package (*www.r-project.org*).

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/3/* .

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

```
mkdir tutorial3; cd tutorial3
```

Create the following symbolic links in the folder *tutorial3*:

```
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 protonated atomistic structure of protein 1CRN to its CG representation:

```
./sirah.ff/tools/CGCONV/cgconv.pl\
 -i sirah.ff/tutorial/3/1CRN.pqr\
 -o 1CRN_cg.pdb
```

Notice: The mapping to CG requires the correct protonation state of each residue at a given pH. We recommend using the PDB2PQR server (*http://nbcr-222.ucsd.edu/pdb2pqr*) and choosing the output naming scheme of AMBER for best compatibility. Be aware that modified residues lacking parameters such as: MSE (seleno MET), TPO (phosphorylated THY), SEP (phosphorylated SER) or others are deleted from the PQR file by the server. In that case, mutate the residues to their unmodified form before submitting the structure to the server.

Notice: Pay attention to residue names when mapping structures from other atomistic force fields or experimental structures. Although we provide compatibility for naming schemes in PDB, GMX, GROMOS, CHARMM and OPLS, there always may be some ambiguity in the residue naming, specially regarding protonation states, that may lead to a wrong mapping. For example, SIRAH Tools always maps the residue name "HIS" to a Histidine protonated at Nε regardless the actual proton placement. Similarly, protonated Glutamic and Aspartic acid residues must be named "GLH" and "ASH", otherwise they will be treated as negative charged residues. In addition, protonated and disulfide bonded Cysteines must be named "CYS" and "CYX" respectively. These kind of situations need to be carefully checked by the users. In all cases the residues preserve their identity when mapping and back-mapping the structures. Hence, the total charge of the protein should be the same at atomistic and SIRAH level. You can check the following mapping file to be sure of the compatibility: *sirah.ff/tools/CGCONV/maps/sirah_prot.map*.

The input file *1CRN.pqr* contains all the heavy atoms composing the protein, while the output *1CRN_cg.pdb* preserves a few of them. Please check both PDB and PQR structures using VMD:

```
vmd -m sirah.ff/tutorial/3/1CRN.pqr 1CRN_cg.pdb
```

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

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

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 1CRN_cg.pdb -o 1CRN_cg.gro
```

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

Notice: By default charged terminal are used but it is possible to set them neutral with option *-ter*

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 1CRN_cg.gro -o 1CRN_cg_box.gro -bt octahedron -d 2.0 -c
```

Add WT4 molecules:

```
genbox -cp 1CRN_cg_box.gro -cs sirah.ff/wt416.gro -o 1CRN_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 1CRN_cg_sol1.gro
```

Topology before editing	Topology after editing

`[ molecules ]`
`; Compound        #mols`
`Protein_chain_A     1`
	`[ molecules ]`
	`; Compound        #mols`
	`Protein_chain_A     1`
	`WT4               756`

Notice: The number of added WT4 molecules (756) may change according to the software version.

Remove WT4 molecules within 0.3 nm of protein

```
echo q | make_ndx -f 1CRN_cg_sol1.gro -o 1CRN_cg_sol1.ndx

grompp\
 -f  sirah.ff/tutorial/3/CPU/em1_CGPROT.mdp\
 -p  topol.top\
 -po delete1.mdp\
 -c  1CRN_cg_sol1.gro\
 -o  1CRN_cg_sol1.tpr
```

```
g_select\
 -f  1CRN_cg_sol1.gro\
 -s  1CRN_cg_sol1.tpr\
 -n  1CRN_cg_sol1.ndx\
 -on rm_close_wt4.ndx\
 -select 'not (same residue as (resname WT4 and within 0.3 of group Protein))'

editconf -f 1CRN_cg_sol1.gro -o 1CRN_cg_sol2.gro -n rm_close_wt4.ndx
```

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

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

```
grep -c WP1 1CRN_cg_sol2.gro
```

Add CG counterions and 0.15M NaCl:

```
grompp\
 -f  sirah.ff/tutorial/3/CPU/em1_CGPROT.mdp\
 -p  topol.top\
 -po delete2.mdp\
 -c  1CRN_cg_sol2.gro\
 -o  1CRN_cg_sol2.tpr

genion -s 1CRN_cg_sol2.tpr -o 1CRN_cg_ion.gro -np 22 -pname NaW -nn 22 -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 the molecular system and check the presence of disulfide bonds. 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 1CRN_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 1CRN_cg_ion.psf 1CRN_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.

Create an index file including a group for the backbone GN and GO beads:

```
echo -e "a GN GO\n\nq" | make_ndx -f 1CRN_cg_ion.gro -o 1CRN_cg_ion.ndx
```

Notice: WT4 and CG ions (NaW, ClW) are automatically set to the group "*SIRAH-Solvent*".

Generate restraint files for the backbone GN and GO beads:

```
genrestr -f 1CRN_cg.gro -n 1CRN_cg_ion.ndx -o bkbres.itp
genrestr -f 1CRN_cg.gro -n 1CRN_cg_ion.ndx -o bkbres_soft.itp -fc 100 100 100
```

When prompted, choose the group "*GN_GO*"

Add restraints to *topol.top*

Topology before editing	Topology after editing

`; Include Position restraint file`
`#ifdef POSRES`
`#include "posre.itp"`
`#endif`
	`; Include Position restraint file`
	`#ifdef POSRES`
	`#include "posre.itp"`
	`#endif`
	
	`; Backbone restraints`
	`#ifdef GN_GO`
	`#include "bkbres.itp"`
	`#endif`
	
	`; Backbone soft restrains`
	`#ifdef GN_GO_SOFT`
	`#include "bkbres_soft.itp"`
	`#endif`

4) Run the simulation

The folder *sirah.ff/tutorial/3/CPU/* contains typical input files for energy minimization (*em1_CGPROT.mdp, em2_CGPROT.mdp*), equilibration (*eq1_CGPROT.mdp, eq2_CGPROT.mdp*) and production (*md_CGPROT.mdp*) runs. Please check carefully the input flags therein.

Make a new folder for the run:

```
mkdir -p run; cd run
```

Energy Minimization of side chains by restraining the backbone:

```
grompp\
 -f  ../sirah.ff/tutorial/3/CPU/em1_CGPROT.mdp\
 -p  ../topol.top\
 -po    em1.mdp\
 -n  ../1CRN_cg_ion.ndx\
 -c  ../1CRN_cg_ion.gro\
 -r  ../1CRN_cg_ion.gro\
 -o     1CRN_cg_em1.tpr

mdrun -deffnm 1CRN_cg_em1 &> EM1.log &
```

Energy Minimization of whole system:

```
grompp\
 -f  ../sirah.ff/tutorial/3/CPU/em2_CGPROT.mdp\
 -p  ../topol.top\
 -po    em2.mdp\
 -n  ../1CRN_cg_ion.ndx\
 -c     1CRN_cg_em1.gro\
 -o     1CRN_cg_em2.tpr

mdrun -deffnm 1CRN_cg_em2 &> EM2.log &
```

Solvent equilibration:

```
grompp\
 -f  ../sirah.ff/tutorial/3/CPU/eq1_CGPROT.mdp\
 -p  ../topol.top\
 -po    eq1.mdp\
 -n  ../1CRN_cg_ion.ndx\
 -c     1CRN_cg_em2.gro\
 -r     1CRN_cg_em2.gro\
 -o     1CRN_cg_eq1.tpr

mdrun -deffnm 1CRN_cg_eq1 &> EQ1.log &
```

Soft equilibration to improve side chain solvation:

```
grompp\
 -f  ../sirah.ff/tutorial/3/CPU/eq2_CGPROT.mdp\
 -p  ../topol.top\
 -po    eq2.mdp\
 -n  ../1CRN_cg_ion.ndx\
 -c     1CRN_cg_eq1.gro\
 -r     1CRN_cg_eq1.gro\
 -o     1CRN_cg_eq2.tpr

mdrun -deffnm 1CRN_cg_eq2 &> EQ2.log &
```

Production (1 μs):

```
grompp\
 -f  ../sirah.ff/tutorial/3/CPU/md_CGPROT.mdp\
 -p  ../topol.top\
 -po    md.mdp\
 -n  ../1CRN_cg_ion.ndx\
 -c     1CRN_cg_eq2.gro\
 -o     1CRN_cg_md.tpr
mdrun -deffnm 1CRN_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/3/* . GPU flags were set for GROMACS 4.6.7, different versions may complain about some specifications.

That's it! Now you can check the simulation using VMD:

Example of trajectory analysis

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

```
trjconv\
 -s   1CRN_cg_em1.tpr\
 -f   1CRN_cg_md.xtc\
 -o   1CRN_cg_md_pbc.xtc\
 -n   ../1CRN_cg_ion.ndx\
 -ur  compact -center\
 -pbc mol
```

When prompted, choose "*Protein*" for centering and "*System*" for output.

Now you can check the simulation using VMD:

```
vmd ../1CRN_cg_ion.psf ../1CRN_cg_ion.gro 1CRN_cg_md_pbc.xtc\
 -e ../sirah.ff/tools/sirah_vmdtk.tcl
```

Calculate the solvent accessible surface (SAS)

Create the following symbolic link in the folder *run/*:

```
ln -s ../sirah.ff/vdwradii.dat
```

Calculate the SAS of the protein along the trajectory:

```
g_sas\
 -s     1CRN_cg_md.tpr\
 -f     1CRN_cg_md_pbc.xtc\
 -n     ../1CRN_cg_ion.ndx\
 -qmax  0              \
 -probe 0.21           \
 -o     area.xvg
```

When prompted, choose "*Protein*" as both the group for calculation and the output.

Notice: The solvent probe radius corresponds to a WT4 bead while a charge of 0*e* refers to any hydrophobic bead. The file *vdwradii.dat* must be placed at the same folder where *g_sas* is executed to assure that the correct van der Waals radii of SIRAH beads are used in the calculation.

You can use Grace to plot the results:

```
xmgrace -nxy area.xvg
```

Visualize the secondary structure

Load the processed trajectory in VMD:

```
vmd ../1CRN_cg_ion.psf ../1CRN_cg_ion.gro 1CRN_cg_md_pbc.xtc\
 -e ../sirah.ff/tools/sirah_vmdtk.tcl
```

At the Tk/Tcl console run the command *sirah_ss* to get the secondary structure of the CG protein.

Notice: After assigning the secondary structure it is possible to represent α-helices with Bendix in VMD 1.9.2 or upper by setting the backbone particle name to GC (do not check the CG box).

To analyze the output files from *sirah_ss*, go back at the shell command line and execute:

```
xmgrace -nxy ss_by_frame.xvg

xmgrace -nxy ss_by_res.xvg
```

The file ss.mtx can be processed to visualize the time evolution of the secondary structure by residue:

```
../sirah.ff/tools/ssmtx2png.R --mtx=ss.mtx

display ssmtx.png
```

The usage of *ssmtx2png.R* can be accessed through:

```
../sirah.ff/tools/ssmtx2png.R --help
```

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: $n_{WT4}=5\frac{n_{ion}}{M}=\frac{5(1)}{0.15}\sim 34$