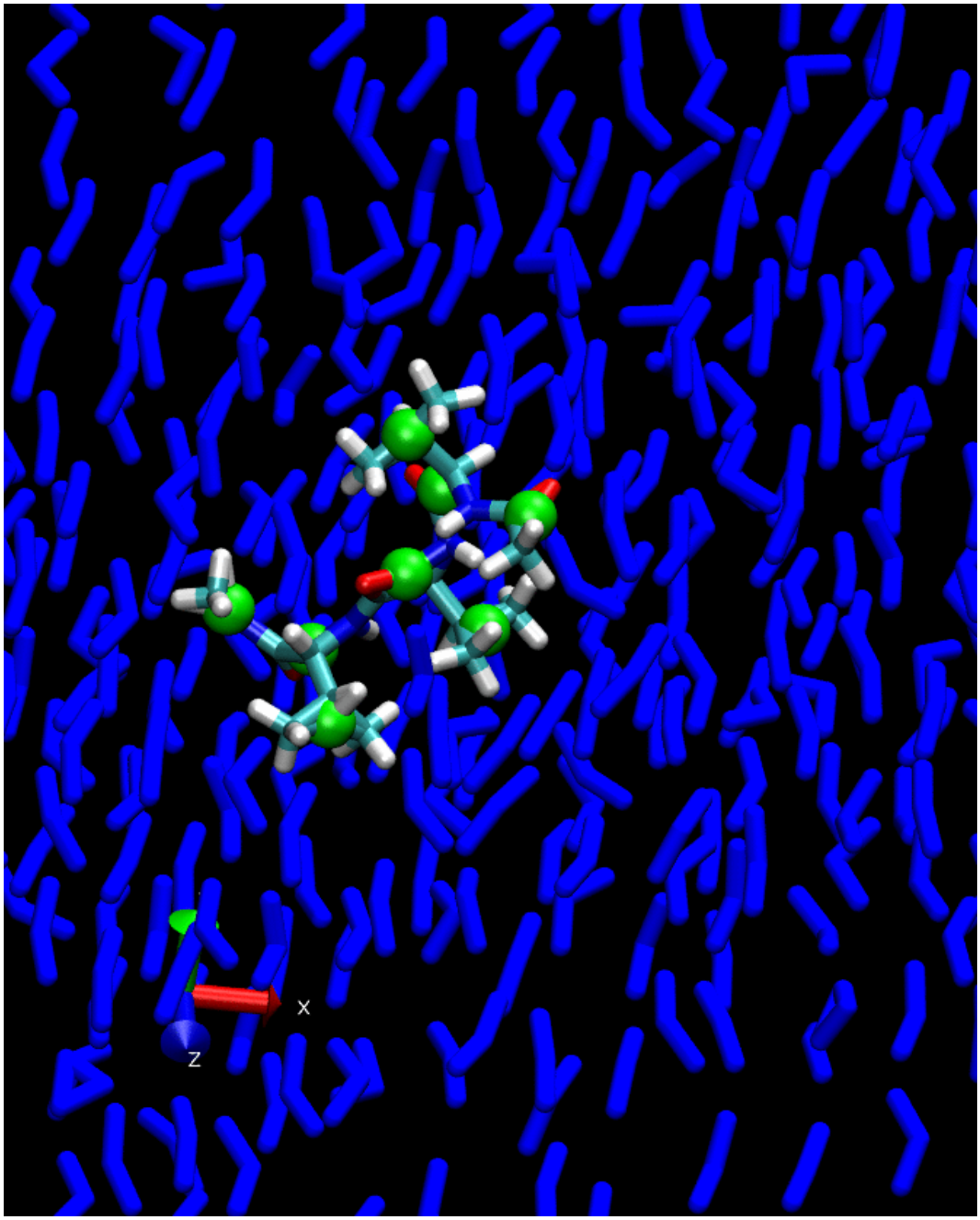
Molecular Simulation Methods with Gromacs



CSC 2016

Alex de Vries with special thanks to Tsjerk Wassenaar

Hands-on tutorial

Multiscaling Simulation and Back-mapping

Aim

Demonstrate how to combine different Classical Mechanical level models within GROMACS and run a HYBRID simulation. In general, this will involve providing user-defined tabulated potentials to GROMACS. A second aim is to demonstrate a back-mapping procedure to build atomistic structures from (partially) coarse-grained models.

Background

Many properties of molecular systems are local, i.e. primarily determined by the molecule and its nearest neighbors. Simulations of small systems therefore often already provide a reasonable to good representation of macroscopic systems. Nevertheless, small system simulations may contain artifacts due to the boundary conditions placed on them. Also, the minimal size of the simulated systems may still prevent sufficient sampling of phase space. The idea of HYBRID models is that a detailed description of relatively distant molecules is not required and that enhanced sampling of a region of interest may be achieved by embedding it in surroundings that represents the environmental influences properly but at a computationally less demanding level. This approach has long been applied to the combination of quantum chemistry and molecular mechanics models (QM/MM), but is much less well developed for combining molecular models at different levels of resolution. It should be kept in mind that developments in this area are still ongoing and that there is as of yet no standard or best practice for this type of simulations.

In this tutorial, we shall focus on the combination of an atomistic (AA, or fine-grained, FG) and a coarse-grained (CG) model. The coarse-grained model is one employing a particle — as opposed to a continuum — description of the surroundings. The standard machinery present in GROMACS allows a quite generic implementation of HYBRID particle-based models through using VIRTUAL sites for the particles at the coarser level in addition to the normal atoms at finer level, and possibly adding user-defined interactions between the particles in accordance with the appropriate expression for the total energy.

The correspondence between the atoms and the virtual sites is called the MAPPING of the atomistic to the CG model. A CG model by itself may use such mapping schemes to parameterize (parts of) the force field. In a purely hierarchical scheme, such as force matching, atomistic simulations are used to calculate the forces on the mapped centers, which are then averaged over many snapshots to define the interactions between the CG particles. In an empirical approach, such as the Martini model, the bonded interactions between CG particles are often refined based on atomistic simulations; the distributions of bond lengths, bond angles and torsional degrees of freedom at the CG level are compared to those from the mapped atomistic simulations and bonded parameters are adapted to get an overall reasonable agreement. The non-bonded interactions in the Martini model are nevertheless generic and based on parameterization to experimental data.

Hybrid OPLS-AA/L—Martini model

The material presented in this tutorial leans on the work originating in the Molecular Dynamics Group at the University of Groningen, in particular that by Tsjerk Wassenaar. The two main publications primarily concern the combination of GROMOS united atom and Martini force fields. Here, we show that one can combine OPLS-AA/L and Martini in the same fashion.

GROMACS implements a considerable number of standard interaction potentials, both bonded and non-bonded. It also enables the users to define their own interaction potentials. In addition, the code implements the generation and use of interaction centers (called VIRTUAL sites) whose positions depend in some geometrically well-defined way on the positions of two or more other particles. A hybrid model combines molecular models at different levels of resolution. The different models may use different types of interaction potentials, and may therefore not be compatible with the same non-bonded (and bonded) functional forms, necessitating a more complex set-up of hybrid model simulations than simulations at a single level of resolution. Here, a set-up will be demonstrated for a combination of the OPLS-AA/L peptide model with the Martini coarse-grained model. The peptide will interact internally at the atomistic resolution, while it interacts with solvent at the coarse-grained (CG) level. The solvent interacts with itself only at CG level. Sections 1-3 take you through setting up and running such a hybrid system.

In Section 4 a tool is introduced to generate fully atomistic solvent configurations from the hybrid simulation that can serve as starting points for production runs at all-atom (AA) level. Such procedures are known as back-mapping techniques.

All files are provided in the tar-ball `csc2016-gradv.tar.gz`, which expands to the main directory `CSC2016/GROMACSADVANCED`. Paths will be given with respect to this directory. A directory with all the results is provided under `WORKED`. Download site(s) for the tar-ball will be given at the workshop.

1. THE MODELS

A. Atomistic model: OPLS-AA/L

As atomistic model we will use the OPLS-AA/L force field.[1] Here, we will have a peptide interacting internally through this force field. The standard atomistic model can be built using the GROMACS tool `pdb2gmx`. This is used to build the model for Lysozyme in the Basic Exercise of this Workshop. Here, we will use a small peptide, trivaline, with methylated terminal ends. A Protein Data Bank (PDB) file can be built very simply using e.g. the builder of `Pymol`, or by finding an existing protein in the PDB with three consecutive valine residues and cutting those out. We wish to use neutral methylated end-groups, normally known in the PDB as ACE for the C-terminal end and NMA for the N-terminal end. To make this work with the standard library file for OPLS-AA/L in GROMACS, the residue name of the N-terminal end must be changed manually from NMA to NAC.

Hands-on

Go to the directory `OPLSAA`. The file `trival.pdb`, built using `Pymol`, is available for you, and it incorporates the naming of the N-terminal end (NAC instead of NMA, done by manual editing) to comply with the GROMACS implementation of the OPLS-AA/L force field. Build the atomistic topology file:

```
gmx pdb2gmx -f trival.pdb -ter
```

This command generates an OPLS-AA peptide topology, `topol.top`, after ENTERING the correct numbers for the options `OPLS-AA/L` force field, `TIP3P` for the water model, and `None` for both terminal ends.

B. Coarse-grained solvent model: Martini

As a solvent, the Martini model for water will be used.[2] This model will also deal with the interactions between the peptide and the solvent (see Section 2 for further details). Standard Martini files can be downloaded (but that is not necessary here because they are provided for you) from the Martini website (`http://www.cgmartini.nl`). They cannot be used as such, however, in the hybrid set-up, and modifications will need to be made. The standard file for the Martini model, which serves as the basis for the modifications can be found in the directory `SOLVENT/martini_v2.1.itp`.

2. THE COMBINATION OF THE MODELS

Models at different levels can be combined in a number of schemes. We describe the simplest approach, viz. we choose to administer all interactions as being at either AA or CG level. The particles making up the peptide will interact at both atomistic and coarse-grained levels. Within the peptide, the interactions should be treated according to the atomistic force field. The atoms of the peptide see the other atoms of the peptide as they would in a normal atomistic simulation. Interactions of the peptide with the solvent are to be CG interactions, i.e. treated entirely according to the Martini model. This clear-cut separation is possible for the system defined here because there are no Coulomb interactions between the AA and CG subsystems. Standard Martini water beads are single, neutral particles. (The proper treatment of electrostatic interactions across the two levels is by no means a finished area of research. Aspects are discussed in Appendix B.)

The present set-up achieves the separation between AA and CG interactions by adding VIRTUAL sites to the atomistic topology, and making sure that these sites have no interaction with the atoms, nor with each other, but only with the solvent. The atomistic topology generated for the peptide must be extended with the virtual sites to build a hybrid topology for the peptide.

A. HYBRID TOPOLOGY FOR THE PEPTIDE

The hybrid topology for the peptide makes use of VIRTUAL SITES. These are described in the GROMACS manual in Section 4.7. In the present implementation, the virtual sites are defined as the centers of mass of a number of atoms. The virtual sites themselves are also particles belonging to the molecule. Thus, two entries need to be ADDED to the atomistic topology. The first is either an extension of the `[ atoms ]` directive or an addtional `[ atoms ]` directive. In both cases, the atom numbering must be consecutive to the atoms already defined. The second extension is a `[ virtual_sitesn ]` directive, which defines how the added particles treated as virtual sites are related to the normal atoms of the molecule. The additional virtual sites and the mapping can be added by hand, but a script is also available to do this for you. The `conf.gro` file of the peptide generated by `pdb2gmx` can be used to generate a Martini topology using an adapted `martinize.py` script (available in `HYBRIDTOPOLOGY` — the original script is downloadable from the Martini web-site) after slight modification of the terminal methylated groups. The standard Martini model does not implement the neutral methylated terminal ends. To distinguish them properly in the `martinize.py` script, a name change of the methyl carbon is required.

Hands-on

Go to the directory `HYBRIDTOPOLOGY`. Copy the standard `pdb2gmx` output file `conf.gro` produced in Section 1 to `conf-martini.gro` and edit the `CH3` of the `ACE` residue to be named `CTJ` and the `CH3` of the `NAC` residue to be named `CTB`. This change is for the benefit of making the adapted `martinize.py` script recognize these neutral methylated termini.

```
cp ../OPLSAA/conf.gro ./conf-martini.gro
[vi/gedit] conf-martini.gro
```

Edit the `conf-martini.gro` to look like this:

```
Gallium Rubidium Oxygen Manganese Argon Carbon Silicon
   60
   20ACE    CTJ    1  -1.244   0.089  -0.561
   20ACE   HH31    2  -1.272   0.187  -0.524
   20ACE   HH32    3  -1.199   0.101  -0.659
   20ACE   HH33    4  -1.333   0.029  -0.571
   20ACE      C    5  -1.146   0.024  -0.467
   20ACE      O    6  -1.176   0.001  -0.349
   21VAL      N    7  -1.012  -0.013  -0.515
...
   23VAL      O   54  -0.103  -0.099  -0.200
   24NAC      N   55  -0.107  -0.003   0.013
   24NAC      H   56  -0.153   0.028   0.100
   24NAC    CTB   57   0.034   0.012  -0.021
   24NAC   HH31   58   0.062  -0.063  -0.094
   24NAC   HH32   59   0.052   0.110  -0.062
   24NAC   HH33   60   0.094  -0.001   0.068
   1.42700   0.79900   0.81020
```

Now, execute the `martinize.py` script:

```
python martinize.py -f conf-martini.gro -ff martini21 \
-multi all -nt -o martini.top -x hybrid.pdb
```

This generates an extension to the molecule definition of the tri-peptide, defining 8 new atom entries that are defined as virtual sites. The flag `-ff` selects the Martini force field version (2.1), the flag `-multi all` produces a topology extension for hybrid AA-CG simulations (omitting this flag, you get a topology for pure Martini CG simulations), the flag `-nt` ensures neutral termini, and the `-x` flag produces a starting structure including the virtual sites. The topology extension is written to the standard file `Protein_A.itp`, which should look like the text shown below:

```
; MARTINI (martini21) Multiscale virtual sites topology section for "Protein_A"
; Sequence:
; JVVVB
; Secondary Structure:
; CCCCC

[ atoms ]
   61   vN0     1   ACE   vBB    61  0.0000 ; C
   62   vP5     2   VAL   vBB    62  0.0000 ; C
   63  vAC2     2   VAL  vSC1    63  0.0000 ; C
   64   vP5     3   VAL   vBB    64  0.0000 ; C
   65  vAC2     3   VAL  vSC1    65  0.0000 ; C
   66   vP5     4   VAL   vBB    66  0.0000 ; C
   67  vAC2     4   VAL  vSC1    67  0.0000 ; C
   68   vN0     5   NAC   vBB    68  0.0000 ; C

;
; Coarse grained to atomistic mapping
;
#define mapping virtual_sitesn
[ mapping ]
   61     2     1     2     3     4     5     6
   62     2     7     8     9    10    21    22
   63     2    11    12    13    14    15    16    17    18    19    20
   64     2    23    24    25    26    37    38
   65     2    27    28    29    30    31    32    33    34    35    36
   66     2    39    40    41    42    53    54
   67     2    43    44    45    46    47    48    49    50    51    52
   68     2    55    56    57    58    59    60
```

Each virtual site is defined as the center of mass of a number of atoms of the tri-peptide. The virtual sites are given first, under the `[ atoms ]` directive, and are numbered starting from 61; the all-atom model tri-peptide has 60 atoms. The atom types given to the virtual sites are those of the Martini model for proteins and peptides, with a prefix `v` (e.g., the standard back-bone bead for a coil residue in Martini is `P5`; therefore the bead-type for the back-bone beads 62, 64, and 66 are `vP5`). The mapping is defined in the directive `[ mapping ]`, which is made to be interpreted by `grompp` as a directive `[ virtual_sitesn ]`. The first number is the index number of the virtual site, the second number (`2`) specifies the type of virtual site (center of mass), and the other numbers are the indices of the atoms that define the virtual site. You may cross-check the mapping with the atom definition given in the atomistic topology.

To weave the two models together, copy the `topol.top` obtained from the atomistic model to `hybrid.top`.

`cp ../OPLSAA/topol.top ./hybrid.top`
`[vi/gedit] hybrid.top`

Use an editor to add the Martini addition in `Protein_A.itp` at the appropriate place, which is just after the end of the definition of the peptide. Excerpts of the file `hybrid.top` are shown below:

```
;     Command line was:
;     pdb2gmx -f trival.pdb -ter
;
;     Force field was read from the standard Gromacs share directory.
;

; Include forcefield parameters
#include "oplsaa.ff/forcefield.itp"

[ moleculetype ]
; Name            nrexcl
Protein             3

[ atoms ]
;    nr        type   resnr residue   atom    cgnr      charge         mass   typeB
chargeB      massB
; residue  20 ACE rtp ACE  q  0.0
     1   opls_135   20   ACE    CH3     1      -0.18     12.011   ; qtot -0.18
     2   opls_140   20   ACE    HH31    1       0.06      1.008   ; qtot -0.12
...
    59   opls_140   24    NAC   HH32   19       0.06      1.008   ; qtot -0.06
    60   opls_140   24    NAC   HH33   19       0.06      1.008   ; qtot 0

[ bonds ]
...

[ pairs ]
...

[ angles ]
...
   59    57    60     1

[ dihedrals ]
...

...
   53    57    55    56     1    improper_Z_N_X_Y


; MARTINI (martini21) Multiscale virtual sites topology section for "Protein_A"
; Sequence:
; JVVVB
; Secondary Structure:
; CCCCC

[ atoms ]
   61   vN0     1   ACE   vBB    61  0.0000 ; C
...
   68   vN0     5   NAC   vBB    68  0.0000 ; C

;
; Coarse grained to atomistic mapping
;
#define mapping virtual_sitesn
[ mapping ]
   61     2     1     2     3     4     5     6
...
   68     2    55    56    57    58    59    60
```

```
; Include Position restraint file
...

[ system ]
; Name
Protein

[ molecules ]
; Compound        #mols
Protein             1
```

B. HYBRID FORCE FIELD FILES

We now have a topology for the hybrid peptide model. In general, further modifications are required before we can run a simulation. First of all, the virtual sites constitute atom-types belonging to the CG model that are not known in the atomistic force field. They must be added to the atom-type list. Of course, the atom-type definitions must not overlap between the two models. For the combination of OPLS-AA/L and Martini this is no problem, but in general, you may need to rename the atom-types of either force field. Also, interactions must be defined between the sets of the two different force fields: either they must be set to zero, or defined explicitly if required. In GROMACS, standard files for interactions are provided for a number of force fields in libraries, provided in directories, see the GROMACS manual, Section 5.8. The OPLS-AA/L force field definitions can be found in the directory `YOUR-PATH-TO-GROMACS/share/gromacs/top/oplsaa.ff` or in `$GMXDATA/top/oplsaa.ff`. We will need to adapt the file `ffnonbonded.itp` in this directory. Since we may not have administrative privileges, we will copy the entire directory to a place that is under our control, effect the required changes there and make sure those files are used in our simulations.

Hands-on

Go to the directory `FORCEFIELDS`, and copy the standard library directory (or use the one already copied for you there):

```
(cp -R $GMXDATA/top/oplsaa.ff .)
cd oplsaa.ff
[vi/gedit] ffnonbonded.itp
```

Valid atom-types for any force field are defined in the file `ffnonbonded.itp`. under the directive `[ atomtypes ]`. In the GROMACS implementation of the OPLS-AA/L force field there is a long list of atom types. For each atom type, the Lennard-Jones sigma and epsilon values are given in the 7th and 8th column of each entry.

```
[ atomtypes ]
; full atom descriptions are available in ffoplsaa.atp
; name  bond_type    mass    charge   ptype      sigma      epsilon
 opls_001   C   6      12.01100     0.500    A    3.75000e-01  4.39320e-01 ; SIG
 opls_002   O   8      15.99940    -0.500    A    2.96000e-01  8.78640e-01 ; SIG
```

The nonbonded interaction parameters can be calculated from the values entered using standard combination rules. GROMACS implements and geometric and arithmetic means, specified in GROMACS in the `[ defaults ]` directive, which can be found in the `forcefield.itp` file. In addition, the interaction parameters for pairs of types can be set explicitly, overriding the standard values. This is done in the `[ nonbond_params ]` directive in the file `ffnonbonded.itp`. Since OPLS-AA/L uses standard combination rules, the `[ nonbond_params ]` directive is missing from the file `ffnonbonded.itp`. You may skip the rest of this page if you like and continue on the next page.

OPTIONAL (intermediate step)

Just by adding the Martini types used for the virtual sites as valid OPLS-AA atom types (make sure you define them as type `V`, not `A`) with standard LJ parameters `0` and `0`, allows you to do a simulation in vacuum with the virtual sites present, but not interacting with anything. You may attempt this yourself by hand (you will need `vN0`, `vP5`, and `vAC2`), or copy the pre-prepared force field files in which this edit has been done for you.

Hands-on

Go to the `VACUUM` directory and either copy the given files:

```
cp -R ../FORCEFIELDS/hyb-vac.ff oplsaa.ff
```

or copy the standard files and edit the file `ffnonbonded.itp`:

```
cp -R $GMXDATA/top/oplsaa.ff .
[vi/gedit] oplsaa.ff/ffnonbonded.itp
```

Next, copy your hybrid starting structure and edited topology there, and you should be able to run a vacuum simulation (using straight cut-off Coulomb interactions instead of PME and without pressure correction):

```
cp ../HYBRIDTOPOLOGY/hybrid.pdb .
cp ../HYBRIDTOPOLOGY/hybrid.top .
gmx grompp -p hybrid.top -c hybrid.pdb -f sd.mdp \
-maxwarn 1
gmx mdrun -v
vmd -e viz-sdrun.vmd
```

When visualizing with `vmd`, using the .`vmd` file, you will see the virtual sites, the positions of the Martini beads as green spheres. Due to the standard settings of `vmd` drawing bonds on the basis of distances, bonds will be drawn between the virtual sites and nearby atoms.

The MARTINI force field has interaction levels that do not follow from a multiplication rule; it therefore defines a full matrix of interactions. The interaction matrix between Martini beads must be added to the nonbonded definition file in the `[ nonbond_params ]` directive of the file `ffnonbonded.itp`. Interactions must be defined between the sets of the two different force fields: either they must be set to zero, or defined explicitly if required. This can be done by hand by appropriately mixing the two files and extending the table with pairs of interactions. Finally, in general one may not want all molecules of a particular kind to interact at a particular level, but only a limited sub-set. For example, we may want to study the active site of a protein atomistically, but embedded in a CG protein and CG water. The virtual sites representing the CG active site must not interact with each other (the atomistic interactions take care of that), but should interact with all surrounding molecules in the appropriate CG manner. One could generate a list of exclusions for the virtual sites while using their normal CG atom types. A more general approach is to double the type of interactions, here CG, where a distinction is made between the types that do not interact with each other, here with prepended `v`, e.g. `vP5` for a `P5`-type particle, and the normal CG types. Now `v`-particles can interact with all CG particles in the normal CG manner, whereas they do not interact at all amongst themselves if the LJ parameters between all `v`-particles are set to zero. This approach is followed here, and you may try to edit an appropriate file yourself, or use a minimal nonbonded file defined for you in `FORCEFIELDS/hybrid.ff/ffnonbonded.itp`. NOTE that the standard Martini force field file available from the Martini website (and provided in the `SOLVENT` directory) uses C6 and C12 parameters instead of sigma and epsilon, as appropriate for the OPLS-AA/L force field. Thus, the Martini parameters must now also be specified in terms of sigma and epsilon (which is the way they are defined in Martini papers).

3. SETTING UP AND RUNNING THE HYBRID SYSTEM

We will now work toward a hybrid simulation of a single atomistic trivaline peptide solvated in Martini water.

A. Surrounding the solute with solvent

First, a starting structure must be prepared in which the peptide is solvated. GROMACS has a standard tool to solvate systems, `gmx solvate`. One can either use an equilibrated solvent box to fill an existing system containing a solute, or insert individual molecules one by one. After one or more molecules are inserted, a check for close contacts is made and new molecules that are too close to existing molecules are removed.

Hands-on

Go to the `HYBRID` directory. Take the hybrid solute co-ordinates, center them in the box and define a box-size of 3.5x3.5x3.5 nm, and fill this with Martini water beads. A co-ordinate file containing 400 equilibrated Martini waters (`water.gro`) is available from the Martini web-site, and is also provided for you in the `SOLVENT` directory.

```
cp ../SOLVENT/water.gro .
gmx editconf -f ../HYBRIDTOPOLOGY/hybrid.pdb \
-center 1 1 1 -box 3.5 3.5 3.5
gmx solvate -cp out.gro -cs water.gro -o solvated.gro \
-radius 0.2
cp ../HYBRIDTOPOLOGY/hybrid.top .
```

Edit the topology file (`hybrid.top`) to include the definition of the Martini water bead (define a `moleculetype` or use a `#include`; the file `SOLVENT/MartiniWater.itp` contains it) and specify the number of water beads just added to your system (right at the bottom in the `[ molecules ]` directive; the Martini water bead is called `W`). The number of water beads added should be clear from the output of the `gmx solvate` command.

```
[vi/gedit] hybrid.top
```

Next, either copy the directory with pre-edited OPLS-AA/L force field files:

```
cp -R ../FORCEFIELDS/hybrid.ff oplsaa.ff
```

or copy the directory with the standard OPLS-AA/L force field files and edit them to include the Martini-bead definitions:

```
cp -R $GMXDATA/top/oplsaa.ff .
[vi/gedit] oplsaa.ff/ffnonbonded.itp
```

OPTIONAL (intermediate stage)

The system thus defined will run; you can test this (in an energy minimization run) by running:

```
gmx grompp -p hybrid.top -c solvated.gro -f em.mdp \
-maxwarn 1
gmx mdrun -v
vmd -e viz-em.vmd
```

B. Generating proper tabulated potentials

Just by making all CG atom types known to the OPLS-AA/L force field and adding the LJ-parameters, the simulation will run, but it is not a proper hybrid simulation because the two force fields do not use the same cut-offs for the nonbonded interactions, nor — in the case of OPLS-AA/L and Martini — the same functional form for the Coulomb and LJ interactions. If you would run an MD simulation with the present set-up, you would not get the density of Martini water.

The next ingredient is thus making sure that the interactions are treated properly according to the appropriate force fields. This may or may not be entirely possible within GROMACS. A powerful mechanism that enables multiscale simulations is the use of the so-called "tables" that GROMACS implements. Internally, potentials and forces are tabulated within GROMACS and the forces are determined by an interpolation scheme between the tabulated values. This is hidden from the user, but users may define their own interaction tables, by which they can input any pairwise nonbonded interaction potential. The next section briefly takes you through the setting up of such tables. They are described in the GROMACS manual, Section 6.10 and a more extensive tutorial is available from the GROMACS website (`http://www.gromacs.org/Documentation/How-tos/Tabulated_Potentials`).

Nonbonded interaction tables can be defined for the interactions between any number of groups. Here, we need:
(i) interaction tables between the atomistic particles, which will be called AA:
— the OPLS-AA/L force field with nonbonded cut-off of 1.0 nm, using a standard LJ potential contained in the file `tables_AA_AA.xvg`,
(ii) interaction tables between coarse-grained particles, which will be called CG:
— the MARTINI force field with nonbonded cut-off of 1.2 nm, using a modified LJ potential, contained in the file `tables_CG_CG.xvg`, and

(iii) interaction tables between the two types (AA and CG) of particles:
— for the example here, the potentials can be set to zero for all distances, but a file with these values must be present. We use the naming convention of GROMACS and let this interaction be the default, which means these interactions must be contained in the file `table.xvg`,

Each tables file contains 7 columns, the first column contains the distances between the pair of particles (in nanometer), and the next six columns implement three generic functions in three pairs of potential and force. Within GROMACS, the magnitude of the interaction or force is determined by the interpolated value taken from the tables file, multiplied by the appropriate Coulomb or LJ prefactors for the pair at hand. The structure of the tables files is thus:

```
distance   f(r)       f'(r)      g(r)        g'(r)       h(r)         h'(r)
```

and with heavily truncated numbers could start like this:

```
0.0000E+00 0.0000E+00 0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00
0.2000E-02 0.4999E+03 0.2499E+06 -0.1562E+17 -0.4687E+20 -0.2441E+33  0.1464E+37
....
```

For the first interaction (modeled on the Coulomb interaction), the prefactor for the functions `f(r)` and `f'(r)` is calculated by multiplication of the two charges and dividing by the dielectric constant (`epsilon_r` in the `mdp`-file). For the second (`g`) and third (`h`) functions, (modeled on the LJ interaction), the prefactors are calculated from the standard combination rules for the C6 and C12 parameters, respectively (which may themselves be calculated from the input sigma and epsilon values), or taken directly from the `[ nonbonded_params ]` list in the `ffnonbonded.itp` file. The distance spacing of the table files can be chosen at will but needs to be uniform. The tables should extend to at least the longest cut-off used in the interactions between the different types of particles, here 1.2 nm, but it is useful to make them extend beyond this — by default, tables are extended 1 nm beyond the cut-off (`mdp` option `table-extension`).

GROMACS provides a number of tables files for you, for example for the Lennard-Jones 6-9 potential, `table6-9.xvg`. They are provided in the `YOUR-PATH-TO-GROMACS/share/gromacs/top` (or `$GMXDATA/top`) directory, with an extension up to 3.0 nm. You will generally need to generate these files yourself, writing a script, a program, etc. Force-matching or related hierarchical coarse-grained force fields[3] generally use tabulated potentials derived from atomistic simulations. For the purpose of this tutorial, the tables files are provided for you in the `FORCEFIELDS` directory and you can simply copy them over. Technical information about the functional forms of the potentials used here is given in the appendix to a paper by Baron et al.[4]. NOTE that you also need a separate `tablep.xvg` file for 1-4 (pair) interactions.

OPTIONAL Hands-on

Go to the `FORCEFIELDS` directory. The tabulated potentials can be viewed using `xmgrace`, for example:

`xmgrace -nxy table_CG_CG.xvg -p xmgr.par`

Return to the HYBRID directory.

A basic `mdp`-file for energy minimization in the hybrid scheme using user-defined potentials is shown below:

```
; RUN CONTROL PARAMETERS
integrator               = steep
nsteps                   = 100
; OUTPUT CONTROL OPTIONS
; Output frequency for coords (x), velocities (v) and forces (f)
nstxout                  = 1
nstvout                  = 1
nstenergy                = 1
; Selection of energy groups
energygrps               = AA CG
; CUTOFF SCHEME
cutoff-scheme            = Group
; NEIGHBORSEARCHING PARAMETERS
; Periodic boundary conditions: xyz, no, xy
pbc                      = xyz
; nblist cut-off
rlist                    = 1.0
; OPTIONS FOR ELECTROSTATICS AND VDW
; Method for doing electrostatics
coulombtype              = user
rcoulomb                 = 1.2
; Relative dielectric constant for the medium and the reaction field
epsilon_r                = 1
; Method for doing Van der Waals
vdwtype                  = user
rvdw                     = 1.2
; Seperate tables between energy group pairs
energygrp_table          = AA AA CG CG
```

Compared to standard settings, points to note in the hybrid set-up are:
(i) definition of the energy groups AA and CG; these are required because their interactions are treated differently, as specified in the energygroup table (see (iii) below):

`energygrps = AA CG`

defines two energy groups, which must be present in the index file (`run.ndx`) supplied by the user (see below);
(ii) specification of user-defined potentials for both coulomb and vanderwaals interactions:

`coulombtype = user`

`vdwtype = user`

tells `grompp` to look for table files;
(iii) designation of which user-defined table is used to handle which interaction:

`energygrp_table = AA AA CG CG`
tells `grompp` to look for `table_AA_AA.xvg` (first pair of arguments, `AA AA`) and `table_CG_CG.xvg` (second pair of arguments, `CG CG`) that define the interactions between the energy groups AA with AA and CG with CG, respectively. For all other interactions the data in the file `table.xvg` will be used.

The separate treatment of interactions is taken care of by defining energy groups as explained above. The energy groups must be known to the program and you must provide an index file to that end, because the energy groups needed are not standard groups known to GROMACS. You can use the conformation with added water beads to build the index file. The sequence shown below does that for you (lines preceded by > are commands within `make_ndx`; the numbering used in this sequence presumes that the standard groups that are defined number up to `13`, where `13` are the `W` beads, and the group `Protein` is group number `1`):

Hands-on (make sure you are in the HYBRID directory)

```
gmx make_ndx -f solvated.gro
> a v*
> 13 | 14
> 1 &! 14
> name 15 CG
> name 16 AA
> q
mv index.ndx run.ndx
```

The next sequence copies the tables files, does a minimization and short MD run in the OPLS-AA/L-Martini hybrid set-up.

```
cp ../FORCEFIELDS/table* .
gmx grompp -p hybrid.top -c solvated.gro -f min.mdp \
-n run.ndx -maxwarn 3
gmx mdrun -v
vmd -e viz-min.vmd
gmx grompp -p hybrid.top -c confout.gro -f md.mdp \
-n run.ndx -maxwarn 2
gmx mdrun -v
gmx trjconv -center -pbc mol -f traj_comp.xtc \
-o viz.xtc < convin
vmd -e viz-run.vmd
```

Hopefully by now, the system is stable enough to run for a longer period of time and you can start production. In my own experience, using constraints leads to problems, and the time-step is fairly limited. Still, there is a substantial reduction in the number of solvent particles, and the overall efficiency in sampling the peptide degrees of freedom is higher than with, say, the TIP3P model as solvent. As a reminder, these types of simulations are at an early stage of development and it is still very much open to debate what methodology is the best one to yield optimal reliability and efficiency. To my knowledge, the combination of OPLS-AA/L and Martini models has not been reported in the literature. Our group in Groningen has published on combining GROMOS united-atom models with Martini models.[5,6] It is clear from that work, that the combination of different resolutions is not without artifacts, and especially the treatment of electrostatics needs careful attention.

4. FINE-GRAINING A SNAPSHOT

Hybrid models promise the best of two worlds: efficient sampling of complex systems and high-level detail in the region of interest. It is nevertheless likely that in practice, some trade-off between accuracy and computational speed will be made causing artifacts near the boundary(ies) of the two (or more) levels. The boundary artifacts may reflect on the region of interest and a method to put in more detail based on the less detailed model can help to study such artifacts. Alternatively, such a method should provide a good starting structure for a simulation at the more detailed level. Approaches that sample long-term processes at coarse-grained level and investigate the details of the interesting intermediate structures after fine-graining have been described already in the literature, and this may well be the most profitable type of multiscaling approaches.

Here, we will take a snapshot from the hybrid OPLS-AA/L peptide in Martini water simulation and put in TIP3P waters based on the positions of the CG water beads. The method also allows fine-graining purely CG snapshots. Several back-mapping schemes have been published. The present method, developed by Tsjerk Wassenaar, is in my view a flexible and user-friendly one.[7]

Back-mapping or reverse coarse-graining or fine-graining a (partially) coarse-grained structure requires a correspondence between the two models; specifically for AA and CG: which atoms make up which bead. Actually, an atom can in principle contribute to several beads. The hybrid set-up described in this tutorial defines virtual sites (the positions of the CG beads) as the centers of mass of a number of atoms. In this mapping scheme, each atom appears only once, but there is no technical reason why it could not appear in the definition of several virtual sites. A back-mapping protocol needs to know at least which atoms contribute to which bead. Existing schemes then use rigid building blocks anchored on the CG bead, or place the atoms randomly near the bead in an initial guess and the structure is relaxed based on the atomistic force field, usually by switching it on gradually. The method presented here allows for an intelligent, yet flexible initial placement of the atoms based on the positions of several beads, thereby automatically generating a very reasonable orientation of the groups of atoms with respect to each other. When back-mapping the hybrid model used in this tutorial we do not need this, because only water beads are fine-grained. The more general procedure is described in Appendix A.

A Martini water bead models four atomistic water molecules. The `backward.py` script used here to fine-grain the hybrid snapshot generates four water molecules around a water bead. By default, any particle named `W` in the structure file is interpreted as a water bead and the script itself generates four 3-atom water molecules (which may then be used as SPC or TIP3P waters). For this exercise, a fully fine-grained starting structure can be prepared with a little editing.

Hands-on

Go to the directory `BACKMAPPING`.

(i) make two copies of a hybrid snapshot

```
cp ../HYBRID/confout.gro W.gro
cp ../HYBRID/confout.gro peptide.gro
```

(ii) remove all non-`W` atoms from the file `W.gro`, edit the second line of the file to correctly state the number of atoms/beads in the file (all particles originally in `W.gro` minus 68)

```
[vi/gedit] W.gro
```

(iii) apply the back-mapping script `backward.py` to the edited file

```
python backward.py -f W.gro -sol -kick 0.2 -o fgW.gro
```

The flag `-sol` tells the script to convert `W` beads into (standard 3-particle water) solvent atoms (four molecules for one bead), and the flag `-kick 0.2` applies random displacements to the initially placed atoms, reducing the strong orientational correlations of the initial placement.

(iv) remove the header (first two lines) and all virtual sites and `W` atoms and last line from the file `peptide.gro`; this file now only contains the real atoms of the peptide

```
[vi/gedit] peptide.gro
```

(v) insert the peptide atoms in the file containing the atomistic water, `fgW.gro`; you should add them near the top of the `.gro` file, just after the line stating the number of atoms in the file and before the solvent molecules. Also edit the number of atoms in the file (add 60, this is the number of atoms in the peptide).

```
[vi/gedit] fgW.gro
```

(vi) copy the atomistic topology file and edit it to add a `SOL` entry in the `[ molecules ]` directive. The number of `SOL` molecules is 4 times the number of `W` beads you had in the hybrid set-up.

```
cp ../OPLSAA/topol.top .
[vi/gedit] topol.top
```

(vii) minimize the back-mapped structure and run a short relaxation/equilibration

```
gmx grompp -p topol.top -c fgW.gro -f min.mdp
gmx mdrun -v
gmx grompp -p topol.top -c confout.gro -f md.mdp \
-maxwarn 1
gmx mdrun -v
gmx trjconv -center -pbc mol -f traj_comp.xtc \
-o viz.xtc < convin
vmd -e viz-run.vmd
```

Now you are all set to run production at fully fine-grained level.

APPENDICES

Appendix A: General back-mapping procedure

The back-mapping of Martini water beads to four 3-site water models (SPC or TIP3P) was employed in Section 4 of the Tutorial. Based on the position of a water bead, positions for the atoms of four water molecules are generated. The coordinates relative to the bead position are in fact hard-coded in the `backward.py` script. The user could change these of course, and the user can use the option `-kick` to randomly disturb the coordinates from their initial positions. Nevertheless, the back-mapped structure of the water clusters is very similar throughout the system. The relaxation time of (bulk) water is fairly short, however, and a short simulation will usually destroy the spatial correlations introduced by the back-mapping procedure.

In the tutorial, back-mapping of the peptide was not necessary because atomistic positions were available from the hybrid simulation. One may envisage situations in which one would like to back-map more complex molecules. One could, for example, add a trivaline peptide to the present system and treat the second trivaline at the coarse-grained level,and run a simulation sampling association-dissociation events. A number of structures may potentially be interesting to study in more detail, for example if different orientations appear to be formed by the monomers in the dimer. In this appendix, the general idea of back-mapping more complex molecules is briefly described. We will illustrate this for trivaline, using as CG coordinates the positions of the virtual sites. More extensive discussion and examples, including tutorial material can be found in the paper by Wassenaar et al.[7] and Supporting Material to that paper.

Hands-on

Go to the directory `BACKMAPPINGA` and copy a snapshot from a hybrid set-up, and edit it such that the atoms (but not virtual sites) of the tri-peptide are removed. This also needs a change in the number of atoms present in the file (second line). Also, the bead names need to be changed, replacing the `v` by a blank space (so `vBB` becomes `BB`, etc. — do not delete `v` because the `.gro` file is supposed to be formatted).

```
cp ../HYBRID/confout.gro cg.gro
[vi/gedit] cg.gro
```

The snapshot is now equivalent to a snapshot that might have been generated in a pure CG Martini simulation of the tri-peptide in water. You can back-map this snapshot to an all-atom tri-peptide + water conformation by using the `backward.py` script similar to the way it was used in Section 4:

```
python backward.py -f cg.gro -sol -kick 0.2 -o fg2.gro \
-to oplsaa
```

You may proceed from here as in Section 4, minimizing, equilibrating, and running a fully atomistic simulation. Perhaps you should have a look at the initial conformation in `fg2.gro` before you do this.

A requirement for the procedure to work, is that the subdirectory `Mapping` contains definitions for how the atomic positions are generated from the CG positions. The subdirectory `Mapping` contains a file for each residue and/or molecule that can be back-mapped, named for the atomistic target force field, e.g. `val.oplsaa.map` for a valine residue targeted to OPLSAA. The structure of map file is explained below for the valine residue. The file `val.oplsaa.map` reads:

```
[ molecule ]
VAL

[ martini ]
BB SC1

[ mapping ]
oplsaa

[ atoms ]
    1     N    BB
    2     H    BB
    3    CA    BB
    4    HA    BB
    5    CB   SC1 BB
    6    HB   SC1 BB
    8   CG1   SC1
    9   HG11  SC1
   10   HG12  SC1
   11   HG13  SC1
   12   CG2   SC1
   13   HG21  SC1
   14   HG22  SC1
   15   HG23  SC1
   16     C    BB
   17     O    BB

[ chiral ]
  CB     CA    N    C
  HB     CA    N    C

[ chiral ]
  HA     CA    N    CB    C ; L-Val
; HA     CA    N    C    CB ; D-Val

[ out ]
  CG2  CB CG1 CA
  HG21 CB CG1 CA
  HG22 CB CG1 CA
  HG23 CB CG1 CA
```

Directives analogous to GROMACS topologies contain specifications that build the atomistic structure from the CG positions. The `[ molecule ]` directive contains the name of the residue or molecule. The `[ martini ]` directive contains the names of the CG beads in the Martini model: valine has two beads called `BB` and `SC1`. The `[ mapping ]` directive contains the name of the object model. Note that this directive may contain multiple object models. If you do not care for the naming convention of different force fields, you can use the same mapping file for the CHARMM36 and OPLS-AA/L force fields, because these are both all-atom models which in the GROMACS implementation also use the same order of the atoms (if not the same names). Thus, the mapping files for the methylated terminal ends explicitly state that they can be used for mapping to both OPLS-AA and CHARMM36 force fields.

The `[ atoms ]` directive contains the index numbers and names of the atoms in the object model and their relation to the CG beads. Note that a single atom may be in a relation with more than one CG bead. The back-mapping procedure starts by putting each atom that is related to a single bead on the position of that bead. If an atom is related to more than one bead, it will be placed on the weighted average position of the beads listed. It is allowed to list the same bead multiple times; thus a line:

```
4  OE1 BB BB BB SC1 SC1
```

places the fourth atom (with name `OE1`) of the residue on the line connecting the `BB` and `SC1` beads at 2/5 of the distance between the beads, starting at the `BB` bead. This mechanism is a simple aid to position atoms already at fairly reasonable starting positions. Using the `-kick` flag displaces all atoms randomly after their initial placement. Note that the script applies a random kick to atoms that are initially put at exactly the same place, e.g. because they are defined by the position of a single bead. Thus, no two atoms will be on top of each other. Switching on an atomistic force field usually results in an error if two interacting atoms are at exactly the same place.

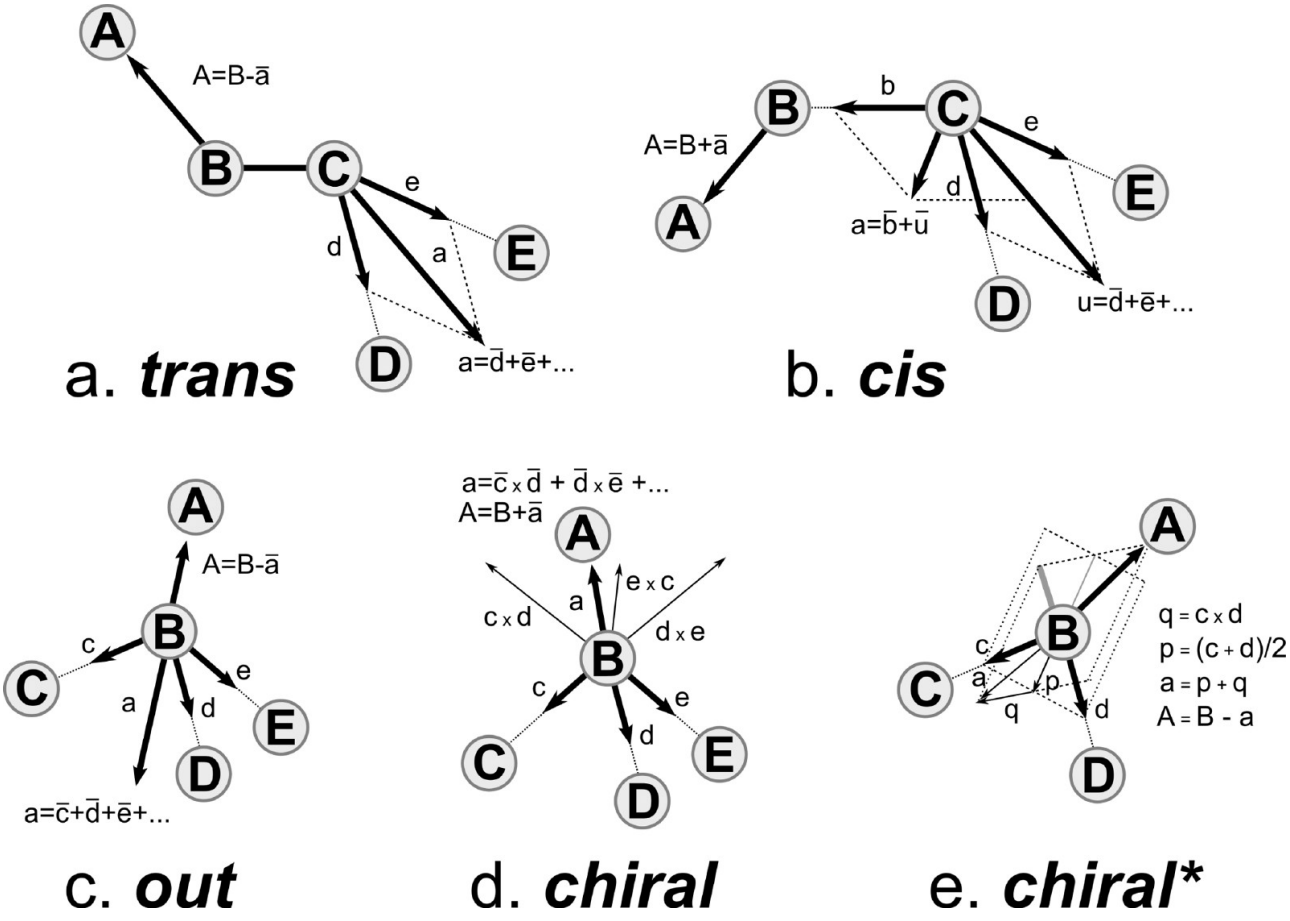


Figure A.1 (Taken from Wassenaar et al., JCTC, 10, 676-690, 2014): Prescriptions for placing atom `A` on the basis of the positions of other atoms (upper case letters `B, C, D, E`). The other atoms define vectors (lower case letters `a, b, c, d, e, p, q`) that are used to calculate the position of `A`. A bar is used over the vectors to denote their normalization. The `x` denotes taking the outer product of two vectors.

The `backward` procedure implements a few other more sophisticated mechanisms to place atoms and some are used in the valine residue. Implementation can be found in the file `Mapping/__init__.py`. The `[ chiral ]` directive generates stereochemically specifically arranged groups of atoms. As is seen for valine, two types of input can be provided. In the first instance of the `[ chiral ]` directive, four atoms are listed on a line. The first atom is the atom to be placed (named `A` in Figure A.1e, ***`chiral*`***) on the basis of the positions of the other three. The Figure shows how vectors are defined from the positions of the other three particles to construct the position of the first atom. In the second instance of the `[ chiral ]` directive, five atoms are listed on a line. This corresponds to the construction shown in Figure A.1d, ***`chiral`***. The `[ out ]` directive may be used to spread out several equivalent atoms (as on a CH3-group) away from the center of the bead in a reasonable manner, as shown in Figure A.1c, ***`out`***. Again, be aware that atoms initially placed on the same spot, are randomly displaced; therefore, using the same reference atoms as in the example still leads to different positions for the generated atoms.

Appendix B: Electrostatics in hybrid simulations

The treatment of electrostatics in hybrid simulations is unlikely to be straightforward, especially if two force fields are combined that are not related to each other in a hierarchical scheme. The two types of force fields might not use the same functional form for the electrostatics. In atomistic models electrostatic potentials are usually based on unscreened Coulomb interactions, either cut off at some distance or treated by full Ewald-type electrostatics in which all charges interact with each other. The standard choice for OPLS-AA/L is Particle Mesh Ewald (PME), a numerical variant of Ewald electrostatics. Coarse-grained models may have no electrostatic term at all, but if they do, the electrostatic potential is usually of some screened form to account for the absence of orientation polarization by dipolar species. In the standard Martini model, the beads modeling water as a solvent do not bear any charge. Formal charges are given to the two beads making up the Zwitterionic lipid head groups, and to the bead(s) representing charged amino-acid side chains. In atomistic models the charge-charge interactions are screened by the explicit water reorientation, which is absent in the CG model. Therefore, the Martini model uses a dielectric constant of 15 to reduce the interaction between the charged species. Also, the Martini model does not use Ewald-type electrostatics; instead the potential smoothly goes to zero at the cut-off distance. It should be noted that the standard Martini model for lipids does run stably under PME and that lipid bilayer properties are similar to those obtained with the standard screened and smoothed Coulomb potential. Thus, combining models may be possible if the electrostatics treatments of the two models can be brought to the same level in a practical sense, possibly with some adjustments of parameters. The published study of combining the GROMOS 53A6 force field with Martini shows that it is not trivial to achieve a working model but at least some fairly reasonable model can be constructed.

This tutorial contains a possible way in which to combine the OPLS-AA/L force field with the polarizable water Martini model using PME electrostatics. In the polarizable Martini model, water consists of three particles, two of which bear opposite charges which are free to move on a circle around the central uncharged bead.[8] The LJ interaction resides on the central bead and the charges within the water bead do not have an electrostatic interaction, but there is an angle potential between the fixed-length bonds from the central particle to each of the charges. The model mimics the orientation polarization of a cluster of four water molecules. The model uses the same form of the Coulomb potential (shifted), but with a relative dielectric constant of 2.5 instead of 15. In the implementation given here, PME electrostatics is used throughout the hybrid system. To account for the difference in dielectric constants (1 in the atomistic model, 2.5 in the Martini model), the tabulated Coulomb potential for the CG-CG interactions and CG-AA interactions are reduced by a factor 2.5 compared to the standard shifted Coulomb potential. Beyond the range of the direct potential, the charges therefore interact with the full and unscreened Coulomb potential, which is not identical to how the interactions are treated in the Martini model, but the differences may be minor. In any case, the model does run stably for the example given here.

Hands-on

Go to the directory `PME`. Study the tabulated potentials and try to run a simulation. You can start from the final snapshot of an earlier simulation. You will first need to convert the water beads to polarizable water beads. This can be done using a python script `triple_w.py` provided on the Martini web-site and present in the directory. To this end, remove all non-W particles from the `.gro` file (here `template.gro`). Paste the co-ordinates of the non-W particles (here saved in `peptide.gro`) in the output file containing the polarizable Martini water co-ordinates (`template_PW.gro`) and adapt the number of particles on the second line. Next, prepare an index file defining CG and AA groups (similar to the one discussed in Section 3B), and finally adapt the topology file to include the definition of polarizable Martini water, changing W to PW.

```
cp ../HYBRID/hybrid.top .
cp ../HYBRID/confout.gro template.gro
cp ../HYBRID/confout.gro peptide.gro
[vi/gedit] template.gro
python triple-w.py template.gro
[vi/gedit] peptide.gro
[vi/gedit] template_PW.gro
gmx make_ndx -f template_PW.gro
mv index.ndx run.ndx
```

Finally adapt the topology file to include the definition of polarizable Martini water, changing W to PW. The definition of the polarizable Martini water can be found in the file `SOLVENT/PolMartiniWater.itp`. You will also need the files defining OPLS-AA/L and MARTINI interactions. You may copy the pre-prepared force field files in which this edit has been done for you. You may attempt this yourself by hand (you will need particle types `POL` and `D` for polarizable Martini water instead of the `P4` for standard Martini water), or copy the pre-prepared force field files in which this edit has been done for you.

```
[vi/gedit] hybrid.top
cp -R ../FORCEFIELDS/hybrid-polW.ff oplsaa.ff
```

Alternatively, you may attempt this yourself by hand (you will need particle types `POL` and `D` for polarizable Martini water instead of the `P4` for standard Martini water). A good place to start would be the set-up for standard Martini water prepared in Section :

```
cp -R ../HYBRID/oplsa.ff oplsaa.ff
[vi/gedit] oplsaa.ff/ffnonbonded.itp
```

Now you should be able to do a minimization and molecular dynamics run.

```
gmx grompp -p hybrid.top -c template_PW.gro -n run.ndx \
-f min.mdp -maxwarn 1
gmx mdrun -v
gmx grompp -p hybrid.top -c confout.gro -n run.ndx \
-f run.mdp -maxwarn 1
gmx mdrun -v
gmx trjconv -center -pbc mol -f traj_comp.xtc \
-o viz.xtc < convin
vmd -e viz-run.vmd
```

The tabulated potentials can be viewed using `xmgrace`, for example:

```
xmgrace -nxy table_CG_CG.xvg -p ../FORCEFIELDS/xmgr.par
```

BIBLIOGRAPHY

[1] G.A. Kaminsky, R.A. Friesner, J. Tirado-Rives and W.L. Jorgensen, Evaluation and Reparametrization of the OPLS-AA Force Field for Proteins via Comparison with Accurate Quantum Chemical Calculations on Peptides, J. Phys. Chem. B 105, 6474-6487 (2001).

[2] S.J. Marrink, H.J. Risselada, S. Yefimov, D.P. Tieleman and A.H. de Vries, The MARTINI force field: coarse grained model for biomolecular simulations, J. Phys. Chem. B 111, 7812-7824 (2007). *

[3] G.S. Ayton, W.G. Noid and G.A. Voth, Multiscale modeling of biomolecular systems: in serial and in parallel, Curr. Opin. Struct. Biol. 17, 192-198 (2007).

[4] R. Baron, D. Trzesniak, A.H. de Vries, A. Elsener, S.J. Marrink and W.F. van Gunsteren, Comparison of thermodynamic properties of coarse-grained and atomic-level simulation models, ChemPhysChem 8, 452-461 (2007). *

[5] A.J. Rzepiela, M. Louhivuori, C. Peter and S.J. Marrink, Hybrid simulations: combining atomistic and coarse-grained force fields using virtual sites, Phys. Chem. Chem. Phys. 13, 10437-10448 (2011).

[6] T.A. Wassenaar, H.I. Ingolfsson, M. Prieß, S.J. Marrink and L.V. Schafer, Mixing MARTINI: Electrostatic Coupling in Hybrid Atomistic–Coarse-Grained Biomolecular Simulations, J. Phys. Chem. B 117, 3516-3530 (2013). *

[7] T.A. Wassenaar, K. Pluhackova, R.A. Bockmann, S.J. Marrink and D.P. Tieleman, Going Backward: A Flexible Geometric Approach to Reverse Transformation from Coarse Grained to Atomistic Models, J. Chem. Theory Comput. 10, 676-690 (2014). *

[8] S.O. Yesylevskyy, L.V. Schafer, D. Sengupta and S.J. Marrink, Polarizable Water Model for the Coarse-Grained MARTINI Force Field, PLOS Comput. Biol. 6, e1000810 (2010).