CSCBP: Exercise 1

Topology Creation & Parameter Transferability

Document version: 16.09.2015

Exercises week 1:	22.09. or 24.09.
Exercises week 2:	29.09. or 01.10.
Deadline for the report:	11.10.

Summary

In this first exercise, you will create a model for ethylene glycol monoacetate, a molecule for which no parameters exist within the GROMOS force field, by combining well-tested GROMOS parameters from similar molecules, ethyl acetate and propanol. This means that you will write most of the corresponding molecular topology file by hand, thereby learning to know the different topological parameters. To assess the quality of your model, you will run simulations of the compound in the liquid state, calculate its density (ρ) and heat of vaporisation (ΔH_{vap}), and compare these with experiment. You will also qualitatively investigate the effect of the environment on the conformational properties of the molecule, by running simulations of the compound in vacuum and in water.

1 Introduction

Any GROMOS simulation using the program `md` (or `md_mpi`) requires at least three files to be provided, namely

- an *input file* (flag `@input`), containing all switches and parameters specifying the desired simulation run (*e.g.* number of steps, timestep, writeout frequency, temperature and pressure coupling, ...),
- a *topology file* (flag `@topo`), containing the specification of the atom content, force-field terms and force-field parameters for the system to be simulated (*e.g.* number and types of atoms, bond, angle, improper and dihedral angle definitions, charges and Lennard-Jones interaction parameters, ...),
- a *starting configuration file* (flag `@conf`), containing the starting coordinates of all atoms and, for a continuation run, the corresponding starting velocities (possibly along with other relevant configurational information).

For this first exercise, all the input and configuration files will be provided to you (but you can still have a look at them for the sake of curiosity!), and the focus will be on the *topology file*. In many situations, creating a GROMOS molecular topology file for your system is relatively easy. This is the case when the GROMOS force field already includes parameters for the molecule of interest or, considering a polymer, for the corresponding monomers. These molecules or monomers are referred to as *building blocks*, and can be assembled using standard GROMOS tools. This will be the situation in Exercise 2. But today, you are not that lucky. You will have to construct your topology file by hand for a new molecule (and perform an initial validation of the resulting model against experimental data).

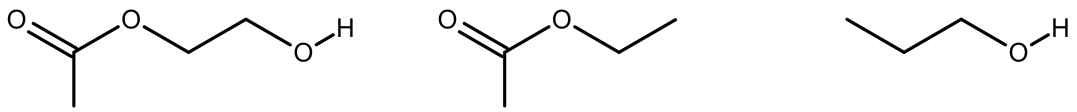
2 Week 1 - Creating the Topology / Starting the Simulations

The goal of this exercise is to create a model for ethylene glycol monoacetate (EGM, see Figure 1), an organic molecule that is in the liquid state at room temperature and ambient pressure. A quick check (*e.g.* using the list of building blocks in the GROMOS manual volume 3) reveals that this specific compound is not available as a ready-made building block in the currently available force fields. On second look, it is not so bad, since two similar compounds, ethyl acetate and propanol (EAE and PPL, see Figure 1), have been parametrised within the GROMOS 53A6$_{\mathrm{OXY}}$ force field [1], which should be a good source for the parameters needed using a *transferability assumption*. Nothing guarantees that the resulting model will be good, *i.e.* it will still need to be validated (and further refined if not sufficiently good). So, we are going to proceed in three steps:

- Construct the topology of EGM by analogy with EAE and PPL, first in terms of "drawings" (Section 2.1).
- Write this topology into a topology file, with a format recognised by GROMOS (Section 2.2).
- Perform an initial characterisation/validation of this model using simulations (Section 2.3).

2.1 Defining the Topology

Your first task is to create the topology for a single EGM molecule. For that, let us start by looking at Figure 1 and determining the different atom types found in the molecule. An *atom type* corresponds to a given atom in a specific chemical environment. This is a relatively fuzzy definition that arises from the observation that one can make "better" force fields by assigning different parameters to the same atom in distinct contexts, as suggested by chemical intuition. For example, the 53A6$_{\mathrm{OXY}}$ force field [1] could only achieve a sufficiently accurate description of (uncharged) oxygen-containing organic compounds by distinguishing three types of oxygen atoms (labelled O, OE and OA, see below). Also remember that for efficiency (and historical) reasons, GROMOS represents an aliphatic carbon atom and the attached non-polar hydrogen atoms as a single "atom" called a *united-atom*.



(a) EGM (b) EAE (c) PPL

Figure 1: Ethylene glycol monoacetate, ethyl acetate and propanol

Given these considerations, EGM is built from the following atom types:

1. C, a plain carbon atom;
2. CH2, a carbon united-atom with two implicitly attached hydrogen atoms;
3. CH3, a carbon united-atom with three implicitly attached hydrogen atoms;

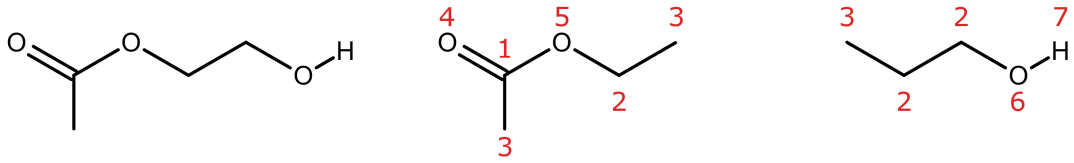
4. O, a carbonyl oxygen atom;

5. OE, an ether or ester oxygen atom;

6. OA, an alcohol oxygen atom;

7. H, an explicit hydrogen atom.

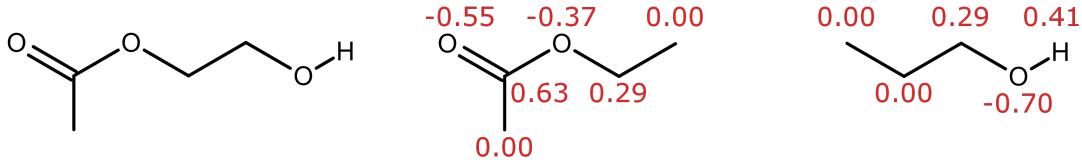
In the following, we will use the above numbering for the integer atom code (`IAC`) of a given atom type[1] (*e.g.* the `IAC` of atom type OE will be 5). The `IAC`s of the atoms of EAE and PPL are shown in Figures 2b and 2c. Now you can (and should!) add the `IAC`s for EGM in Figure 2a.



(a) EGM (b) EAE (c) PPL

Figure 2: Atom Types (`IAC`)

By choosing an `IAC` for each atom in the molecule, you have actually decided the form of the van der Waals interaction between any pair of atoms (short-range repulsion when the atoms overlap + longer-range attraction due to dispersion). This is because GROMOS determines the parameters of this interaction based on the `IAC` of the two interacting atoms[2]. The electrostatic (Coulombic) interaction, however, is not determined by the `IAC`. For this one, you have to assign partial charges to all atoms in your molecule. Because we have partial charges for EAE and PPL from the 53A6$_{\mathrm{OXY}}$ force field, we will simply assume here that these charges can be directly transferred to EGM by analogy. The partial charges of the atoms of EAE and PPL are shown in Figure 3b and 3c. Now you can (and should!) add the partial charges for EGM in Figure 3a.



(a) EGM (b) EAE (c) PPL

Figure 3: Atomic Partial Charges

In a next step, the bond stretching, bond-angle bending, improper dihedral-angle distortion and proper dihedral-angle torsion potential energy terms defining the covalent flexibility of the molecule have to be defined. The first three types of terms are defined by a reference value (length or angle) and a force constant regulating the strength (the energetic cost of a deviation away from

[1]The 53A6$_{\mathrm{OXY}}$ force field considers many more molecules than just EAE and PPL, and has thus as many as 53 atom types. However, for simplicity, we have taken the types relevant for EGM and renumbered them from 1 to 7.

[2]This is not entirely true. Within a molecule, you still need some additional information on the topological relationship between the pair, *i.e.* we need to know the `IAC`s of the two interacting atoms and whether they are in a normal, third-neighbour or excluded relationship.

the reference value). The torsional terms are periodic and characterised by a multiplicity and a phase shift instead of a single reference value.

These parameters can be obtained from experimental structure-determination (*e.g.* X-ray or NMR) and spectroscopic (*e.g.* IR) measurements, or from quantum mechanical (QM) calculations. Fortunately, the parameter optimisation was already performed for EAE and PPL by the authors of the 53A6$_{\mathrm{OXY}}$ force field, so that we will simply assume that they are transferable by analogy to EGM. The covalent parameters of the 53A6$_{\mathrm{OXY}}$ force field relevant for EGM are listed in Table 1. For each of the four types of term, each set of parameter is assigned a type code for the ease of further reference.[3,4] The type codes of the bond stretching, bond-angle bending, improper dihedral distortion and proper dihedral torsion terms for EAE and PPL are shown in Figures 4b - 7b and Figures 4c - 7c, respectively[5]. Now you can (and should!) add the corresponding type codes for EGM in Figures 4a - 7a.

2.2 Writing the Topology File

Congratulations! You have now determined all potential energy terms present in the EGM molecule, *i.e.* the form of the van der Waals, electrostatic and covalent interactions. Time to let GROMOS know about this success. And for this, you have to encode your freshly acquired knowledge into a *topology file*.

If you have not done so yet, copy the main exercise directory `ex1/`[6] to your home directory. See Appendix A for an overview of the files found in the directory.

Open the file `topo/EGM.top` in your favourite text editor. In general, a GROMOS file consists of a series of *blocks*. Each block starts with the block name in capital letters and ends with the capitalised keyword END, both required to be on separate lines. Each block must contain a well-defined number of entries (character strings, integer numbers or real values, depending on the specific block) in a defined order. It is important to prepare GROMOS files with great care, as input errors can only be detected in some cases. In the file `EGM.top`, you will find a sketch for the topology file of EGM based on the model you constructed in Section 2.1. All the required blocks are listed, but some have been left empty, with the line `# TODO` instead of a content.[7] Your task is to fill these blocks yourself.

The subsections 2.2.1-2.2.4 below describe in more details the block content of the file (and what you have to place into the blocks), from top to bottom. Subsection 2.2.5 tell you how to do a first consistency check of the file you created using the GROMOS program `check_top`.

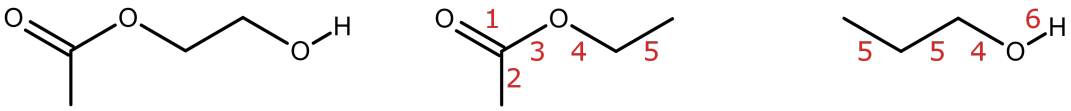
[3]Here again, the 53A6$_{\mathrm{OXY}}$ force field considers many more molecules than just EAE and PPL. For simplicity, we have only taken the types relevant for EGM and renumbered them from 1.

[4]Note that GROMOS can use two different forms of potential-energy terms for both bond stretching and bond-angle bending (the form employed in a given simulation is selected in the input file). For the bond stretching, the quartic and harmonic force constants are listed with a "q" and a "h" superscript, respectively. For the bond-angle bending, the cosine-harmonic and angle-harmonic force constants are listed with a "c" and a "h" superscript, respectively.

[5]The proper dihedral torsion potential number 1 is defined using the atoms (CH3)-(C)-(OE)-(CH2).

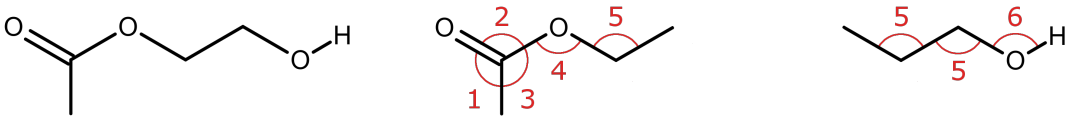
[6]See the document "CSCBP computational setup" for the exact location of the directory

[7]Note that in a GROMOS file, any line starting with a hash (`#`) is a comment, *i.e.* the entire line ignored upon reading. Feel free to write your own explanatory comments when making your topology file, it is actually a good practise.



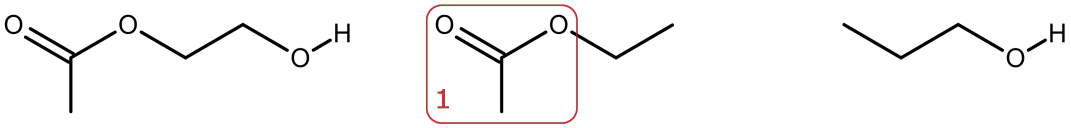
(a) EGM (b) EAE (c) PPL

Figure 4: Bond Stretching Potentials



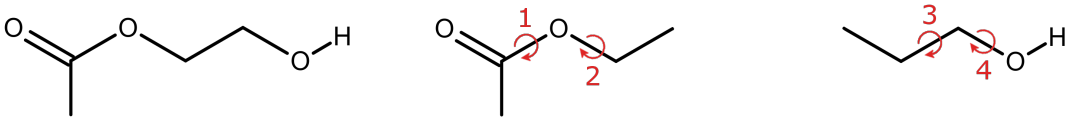
(a) EGM (b) EAE (c) PPL

Figure 5: Bond-Angle Bending Potentials



(a) EGM (b) EAE (c) PPL

Figure 6: Improper Dihedral Distortion Potentials



(a) EGM (b) EAE (c) PPL

Figure 7: Proper Dihedral Torsion Potentials

bond type code	K_b^q /[kJ mol^{-1} nm^{-4}]	K_b^h /[kJ mol^{-1} nm^{-2}]	$b_\circ$ /[nm]
1	1.66×10^7	5.02×10^5	0.123
2	7.15×10^6	3.35×10^5	0.153
3	1.02×10^7	3.77×10^5	0.136
4	8.18×10^6	3.35×10^5	0.143
5	7.15×10^6	3.35×10^5	0.153
6	1.57×10^7	3.14×10^5	0.100

(a) Bond Stretching Potential

angle type code	K_θ^c /[kJ mol^{-1}]	K_θ^h /[kJ mol^{-1} deg^{-2}]	$\theta_\circ$ /[deg]
1	750	0.153	125.0
2	700	0.153	122.0
3	545	0.140	113.0
4	635	0.153	117.0
5	530	0.140	111.0
6	450	0.122	109.5

(b) Bond-Angle Bending Potential

improper type code	K_ξ /[kJ mol^{-1} deg^{-2}]	$\xi_\circ$ /[deg]
1	0.0510	0.0

(c) Improper Dihedral Distortion Potential

dihedral type code	K_ϕ /[kJmol^{-1}]	δ /[deg]	m
1	16.70	180.0	2
2	3.77	0.0	3
3	5.92	0.0	3
4	1.26	0.0	3

(d) Proper Dihedral Torsion Potential

Table 1: Potential Energy Terms in the 53A6$_{\mathrm{OXY}}$ force field

2.2.1 Title, Constants, Atom and Residue Names

First comes the TITLE block. Here, you can give a title consisting of any number of lines. This is merely a help for yourself, to remember what kind of topology you are doing, very useful in case you are working with dozens of topologies at a time or have to find out half a year later what exactly you were doing in this file (the topology title will also be printed in the md output, so you know what topology was used for a specific run). Type a descriptive title replacing the # TODO tag, so that your block looks something like

```
TITLE
    My first GROMOS topology
    Compound: Ethylene glycol monoacetate (EGM)
    Author: Sir Isaac Newton
    Note: Adapted from ethyl acetate and propanol in 53A6_OXY
    Date: September 22, 2015
END
```

The next two blocks are already filled and need no change. The first one, PHYSICALCONSTANTS, is rather self-explanatory and sets a number of physical constants. The second one, TOPVERSION, is used internally by the GROMOS program package to keep track of different topology versions.

The following block ATOMTYPENAME contains the number of distinct atom types in your topology[8] (first line), then lists their name in order of ascending IAC (each on a separate line). For simplicity, we already filled out this block for you. You will recognise the 7 atom types of Section 2.1, plus an additional oxygen atom called OW and given the IAC of 8. It will be used for the representation of water.

The RESNAME block contains the number of residues (monomers) in the molecule, then lists their name in order of ascending residue number. For simplicity, we already filled out this block for you. Your molecule is not a polymer, so it has only one "residue" which we called EGM. This string will be used to refer to the residue in the analysis programs, but has no influence on the parameter selection.

2.2.2 The SOLUTEATOM Block

The SOLUTEATOM block contains the number of atoms in the molecule (first line), then lists the atoms in sequence and provides information on a per-atom basis (two lines per atom[9]). For simplicity, we already filled out this block, but you should definitely go through it very carefully and understand what is there. For each atom, the two lines must list in sequence:

- ATNM: The atom number.
  This number will be used to define the covalent potential-energy terms (*e.g.* in the list specifying the bonded atom pairs). The numbers of the different atoms should start from

[8] In general, this block will contain not only the types you need for your specific molecule, but the entire set for a given force field, *e.g.* 53 types for the $53A6_{OXY}$ force field. This is because we do not want to renumber the atom types for every new molecule. We better use one numbering for all atom types in a force field, take the types we need for the given molecule, and ignore the others. For the present exercise, however, we did the renumbering and only include the 7 useful atom types plus one needed for water.

[9] It would also work to make these two lines a single one (GROMOS moves to the next atom whenever it has all records for one atom, irrespective of the number of line breaks), but the two-line format is more readable.

one and be incremental in the list[10]. For your molecule, the numbering we chose is shown in Figure 8a.[11]

- `MRES`: The residue number.
  This number specifies the residue to which the atom belongs. In our case, having only one residue, this will always be `1`.

- `PANM`: The atom name.
  This string will be used to refer to the atom in the analysis programs, but has no influence on the parameter selection[12]. For your molecule, the naming we chose is shown in Figure 8b.

- `IAC`: The integer atom code.
  The `IAC` is used to define the (Lennard-Jones) interaction parameters associated with this atom. These parameters are given later in the form of a two-dimensional matrix, the lines and columns of which are the `IAC`s of the two interacting atoms (see `LJPARAMETERS` block later). The `IAC`s listed in the block should match those you have in Figure 2a.

- `MASS`: The atomic mass, given in u.
  Watch out that the masses of the united-atoms include those of the attached hydrogen atoms.

- `CG`: The atomic partial charge of the atom, given in e.
  The charges listed in the block should match those you have in Figure 3a.

- `CGC`: The charge group code (boolean, `0` or `1`).
  For different reasons[13], sets of successive atoms are grouped into so-called charge groups (CG). A `0` indicates that the current CG starts or continues. A `1` indicates the last atom of the CG. So, your molecule has 3 CG. Try to draw them in Figure 3a. What are their net charges?

- `INE`: The excluded-atom list.
  Excluded-atom pairs are pairs of close covalent neighbours in a molecule (*i.e.* normally those separated by one or two bonds) which are exempted from mutual van der Waals and electrostatic interaction. For a given atom, the `INE` record specifies the number of atoms with higher `ATNM` sequence numbers that belong to the exclusion list of this atom, then lists their `ATNM` sequence numbers in ascending order. Are the excluded pairs listed in the file those you would expect considering the structure of your molecule?

- `INE14`: The third-neighbour list.
  Third-neighbour atom pairs are pairs of third covalent neighbours in a molecule (*i.e.* those separated by three bonds), that are subject to special (reduced) mutual van der Waals interaction. The electrostatic interaction is unchanged. For a given atom, the `INE14` record specifies

[10] Note that although the list in the file must be sequential, there are still many different ways to number the atoms of a molecule (some choices make your life easier than others, though).

[11] Warning: Do not confuse the atom number `ATNM` in Figure 8a with the atom type `IAC` in Figure 2a. The first one is just a "label", the second one determines the van der Waals interactions.

[12] Here also, there are many different ways to name the atoms of a molecule (some choices make your life easier than others, though).

[13] These are explained in the lecture: computational speed-up (making a list of CG pairs is faster than making a list of atom pairs) and reduced cutoff noise (cutting off interactions between neutral CG induces less noise than cutting off interactions between charged atoms). The CGs should be reasonably small and as much as possible overall neutral.

the number of atoms with higher ATNM sequence numbers that belong to the third-neighbour list of this atom, then lists their ATNM sequence numbers in ascending order. For readability, one generally aligns the INE14 record just below the INE record. Are the third-neighbour pairs listed in the file those you would expect considering the structure of your molecule?

(a) ATNM (b) PANM

Figure 8: The unique atom numbers (ATNM) and the atom names (PANM), as defined in the SOLUTEATOM block. Not to be confused with the atom type (IAC), given in Figure 2a.

2.2.3 Covalent Potential Energy Terms

The BONDSTRETCHTYPE block defines all possible bond stretching potential energy terms in the molecule. The first entry defines the number of distinct bond potential energy terms, the following values, in groups of three, the actual potential energy terms. Each potential definition consists of the quartic (CB) and the harmonic (CHB) force constants and the bond length at the energy minimum (B0). The successive bond potentials are referred to by a bond-type code (in ascending order starting from 1), which will be used later to assign a given bond-stretching potential to a given bond in the molecule.
To complete the blocks defining the bond stretching potentials, do the following:

- Carefully compare the values from Table 1 with the entries in the BONDSTRETCHTYPE block to make sure that the order and the definitions are identical.
- Bring together the bond definitions (BONDSTRETCHTYPE block and Figure 4a) and the atom numbers (SOLUTEATOM block and Figure 8a) to fill in the BONDH (all bonds containing at least one hydrogen atom) and BOND (all other bonds) blocks.[14] The first entry denotes the number of bonds in the block, the entries afterwards define the bonds. Every bond definition consists of three integer numbers, the first two referring to the atom numbers defined in the SOLUTEATOM block (ATNM, *not* IAC), the third defining the bond type as implicitly defined by the order in the BONDSTRETCHTYPE block.

For a three-atomic molecule containing no explicit hydrogen atoms, the three blocks could look like

```
BONDSTRETCHTYPE
#  NBTY: number of covalent bond types
    2
#  CB:  quartic force constant
#  CHB: harmonic force constant
#  B0:  bond length at minimum energy
```

[14]The differentiation has purely historical reasons.

```
#          CB          CHB          B0
         1.00e7       5.21e5       0.110
         2.32e7       7.42e5       0.155
END
BONDH
#  NBONH: number of bonds involving H atoms in solute
    0
#  IBH, JBH: atom sequence numbers of atoms forming a bond
#  ICBH: bond type code
#   IBH    JBH ICBH
END
BOND
#  NBON: number of bonds NOT involving H atoms in solute
    2
#  IB, JB: atom sequence numbers of atoms forming a bond
#  ICB: bond type code
#    IB      JB  ICB
     1       2     1
     2       3     2
END
```

In analogy to the bond-stretching potential, fill out the bond-angle bending potential blocks (`BONDANGLEBENDTYPE`, `BONDANGLEH`, `BONDANGLE`), the improper dihedral distorsion blocks (`IMPDIHEDRALTYPE`, `IMPDIHEDRALH`, `IMPDIHEDRAL`) and the proper dihedral torsion blocks (`TORSDIHEDRALTYPE`, `DIHEDRALH`, `DIHEDRAL`). Note that for each kind of potential, it is again distinguished whether the potential includes a hydrogen atom or not. In contrast to the bond-stretching terms, which involve two atoms, the bond-angle terms involve three atoms and the (proper and improper) dihedral terms involve four atoms. The order of the bond-angle terms (first atom - central atom - last atom) and of the proper dihedral terms (first atom - central atom 1 - central atom 2 - last atom) follows the bonds between the atoms. The bond-angle and proper dihedral terms are identical under inversion of the definition order. For the (planar or tetrahedral) improper dihedral terms, the central atom is defined first, followed by the three outer atoms in arbitrary order.

The next (already filled) blocks in the file are the `CROSSDIHEDRALH` and `CROSSDIHEDRAL`, defining cross dihedral potential energy terms not used in our molecule. This completes the covalent interaction part of the topology.

2.2.4 Lennard-Jones Parameters, Temperature and Pressure Groups, Exceptions, Solvent

The `LJPARAMETERS` block spans the (triangular) two-dimensional matrix defining the Lennard-Jones interaction between all defined atom types, both for the normal as well as for the third-neighbour interactions. In case of nonbonded interactions, the simulation program will resort to this matrix to determine the correct potential parameters[15].

[15]Generally, first and second covalent neighbours (connected by one or two bonds) are excluded and will have no van der Waals interaction at all. Second covalent neighbours (connected by three bonds), *i.e.* third-neighbours, will have reduced van der Waals interactions. Only beyond is the interaction "normal" and specified by the `IAC`s of the

The SOLUTEMOLECULES block is used to subdivide the solute into separate molecules or fragments, if necessary. Here, we have only one molecule defined. The last atom of each molecule or fragment (or of the only molecule) must be given.

The next blocks, TEMPERATUREGROUPS and PRESSUREGROUPS allow to treat certain atoms differently in terms of the coupling to a thermostat or a barostat - you will hear about these techniques later in the lecture. For now, we will simply treat all atoms equivalently, which means defining only one temperature and pressure group and give the last atom of the molecule as last atom of the group, just as with the SOLUTEMOLECULES block above.

The LJEXCEPTIONS block is another special block allowing to tweak certain pairwise Lennard-Jones interactions specifically. We do not need this for our molecule.

The last two blocks of the topology do not concern the solute anymore, but the solvent it is (or might be) solvated in. The use of solvent can still be turned off in the input file, even though it needs to be present here. Here we chose to have water as a solvent, represented by the SPC (simple point charge) model [2], a very simple but surprisingly accurate water model represented by three point charges. The SOLVENTATOM block defines the properties of the solvent atoms, starting by a unique identifier (I), followed by a name (ANMS), which in analogy to the solute is only used for the identification during analysis. Then, the Lennard-Jones parametrisation is given *via* the IACS code, as well as the mass (MASS, in u) and the (point) charge (CGS, given in e). No exclusion or exception list is provided, as all intramolecular non-bonded interactions in solvents are excluded by definition.

The last block, SOLVENTCONSTR, gives the solvent constraints, defining the geometry of the solvent molecules.[16]

2.2.5 Final Check

To help finding mistakes in hand-written topologies, GROMOS++ offers a tool called check_top. After you saved your topology written in the Sections 2.2.1 to 2.2.4, navigate to your ex1 directory, and invoke check_top by calling

```
check_top @topo topo/EGM.top @coord crd/EGM.g96 @pbc r gbond
```

topo/EGM.top is the topology file you just wrote, crd/EGM.g96 a single EGM molecule coordinate file provided, and @pbc r gbond tells the program about the boundary conditions and the gathering. Please refer to the doxygen for further information on the options.

In a first phase, check_top performs a number of simple consistency checks on your topology and writes out errors in case it finds something dubious. Read the output carefully and try to find possible mistakes in your topology. A correct topology passes the test without errors or warnings!

In a second phase, check_top calculates the potential energies of all bonded potentials defined in the topology using the coordinates provided with the @coord flag. An unusually high energy suggests that the coordinates are far from the reference position given by the potential assigned and

two interacting atoms.

[16] Solvents in GROMOS do not have potential energy terms for bonds, bond angles and dihedrals, but are kept completely rigid. You can imagine a water molecule as a triangle with three bonds, each of them always kept at the same distance. This is computationally much more efficient (as constraints are more efficient than bond potentials, and there are a lot more solvent molecules than solute molecules in a typical system), and the approximation done by this treatment is justifiable, since we are in general interested in an exact treatment of the solute under the effect of a solvent, not in the exact treatment of the solvent. If a flexible solvent is needed, it must be defined as a set of molecules formally belonging to the solute part of the topology.

indicates a probable error in the topology, given the coordinates are correct. Use this second indication to further check your topology. As a guideline, the covalent potential energies corresponding to the given coordinates should be around $0.46\,\mathrm{kJ\,mol^{-1}}$.

2.3 Initial Validation

Up to now, we have created a model for EGM inspired by analogy with existing parameters for EAE and PPL, and managed to encode this information into a GROMOS topology file. This is expected to be a reasonable way of proceeding if the EAE-like and PPL-like fragments of EGM do not present a dramatic influence on each other in terms of electron density distribution.[17] Even if we believe the transferability assumption to work well in the present case, we cannot trust our model without a minimal amount of *validation* against experimental data.

The following three subsections describe in turn: (*i*) the principle of the validation; (*ii*) the setup of the required simulations; (*iii*) the execution of these simulations on the beaver cluster.

2.3.1 Principle of the Validation

For the purpose of initial validation, we are going to simulate EGM under standard conditions, *i.e.* at 298.15 K and 1 bar, considering three distinct situations:

- GAS: In the gas phase (ideal gas limit[18])
- LIQ: In the pure-liquid phase
- WAT: In water (infinite dilution limit)

For the GAS, we consider a set of N EGM molecules placed at very large distances from each other under periodic boundary conditions in a box large enough to mimic an ideal (non-interacting) gas. For the LIQ, we consider a set of N (the same number, for simplicity) EGM molecules within a cubic box simulated under periodic boundary conditions and kept at a size yielding the appropriate pressure for the chosen (standard) conditions. For the WAT, we consider a single EGM molecule surrounded by N_{wat} water molecules within a cubic box simulated under periodic boundary conditions again kept at an appropriate size.

The LIQ simulation will give access to the pure-liquid density (ρ) using the equation

$$\langle\rho\rangle = \frac{MN}{\langle V\rangle}\,, \tag{1}$$

where M is the molecular weight of EGM, N the number of molecules in the computational box, V the volume of the computational box, and $\langle\cdot\rangle$ denotes an average over the simulated trajectory.

[17] A lot of organic chemistry is about substituent effects, *i.e.* cases where this assumption breaks down! For example, the properties of a substituted benzene ring are seldom the "sum" of those of a free benzene and a separate substituent, due to resonance effects.

[18] Thermodynamically, an ideal gas can exist at any temperature and pressure, unlike a liquid or a solid, which are real phases bound to a specific part of the phase diagram. In practice, we will mimic an ideal gas by placing molecules so far from each others that there is no interaction, keeping the (very large) box volume unchanged during the simulation. The pressure during the simulation will not be properly defined, but is not relevant for the value of the (intramolecular) gas phase energy that we are interested in.

The LIQ and GAS simulations will give access to the enthalpy of vaporisation (ΔH_{vap}) of the liquid using the equation

$$\Delta H_{\text{vap}} = \frac{\langle \mathcal{U} \rangle_{\text{gas}}}{N} - \frac{\langle \mathcal{U} \rangle_{\text{liq}}}{N} + RT\,, \tag{2}$$

where $\mathcal{U}$ is the total potential energy, R the ideal gas constant, T the absolute temperature, and $\langle \cdot \rangle_{\text{gas}}$ and $\langle \cdot \rangle_{\text{liq}}$ denote averages of the GAS and LIQ trajectories, respectively.

A third quantity of great interest for validation is the hydration free energy ΔG_{wat} of the molecule. Calculating this quantity is, however, a bit more complicated (as you will learn in Exercise 5), and in the WAT simulation, we are merely going to simulate the compound in water to examine its conformational properties.

The experimental data for ρ and ΔH_{vap} is reported in Table 2, along with that for the liquids EAE and PPL. There, you can see in particular that the 53A6$_{\text{OXY}}$ force field does a good job at reproducing experimental data for EAE and PPL. The question is: does our newly constructed model for EGM do comparably well?

comp	$\rho^{(\text{e})}(298.15\,\text{K})$ /[kg m^{-3}]	$\rho^{(\text{s})}(298.15\,\text{K})$ /[kg m^{-3}]	$\Delta H^{(\text{e})}_{\text{vap}}(298.15\,\text{K})$ /[kJ mol^{-1}]	$\Delta H^{(\text{s})}_{\text{vap}}(298.15\,\text{K})$ /[kJ mol^{-1}]	T_b /[K]	$\Delta H^{(\text{e})}_{\text{vap}}(T_b)$ /[kJ mol^{-1}]
PPL	800 [4]	781.0±0.15 [1]	47.5 [4]	49.5±0.02 [1]	370 [3]	41.44 [3]
EAE	895 [4]	881.8±0.27 [1]	35.6 [4]	36.0±0.02 [1]	350 [3]	31.94 [3]
EMG	1108 [3]		63.9 [5]		461 [3]	55.1 [5]*

Table 2: Experimental data ($^{(\text{e})}$ superscript) and simulation results using the 53A6$_{\text{OXY}}$ force field ($^{(\text{s})}$ superscript) for the compounds considered. The boiling temperature T_b and the enthalpy of vaporisation at the boiling point $\Delta H^{(\text{e})}_{\text{vap}}(T_b)$ are given as an additional information. They are not needed for our simulations but listed in view of Question 5 in Section 4.3.
*No experimental value at boiling point could be found. The value reported here is measured at 378 K.

2.3.2 System Size, Combined Topologies

The topology we have created contains a single solute molecule. As explained in Section 2.3.1, we need to simulate a larger number N of solute molecules for the GAS and LIQ calculations. For the WAT calculation, we need only one solute molecule, but surrounded by N_{wat} water (solvent) molecules. To chose the number of molecules, we go for a simple rule: The system should be big enough to be simulated under periodic boundary conditions with a cutoff of 1.4 nm[19] without ever encountering self-interactions with a molecules periodic copy, but otherwise as small as possible to keep simulations short. Values which have proved to work well for similar systems are $N = 512$ for the GAS and LIQ simulations and $N_{\text{wat}} = 1024$ for the WAT simulations.

To be able to use 512 solute molecules in a GROMOS simulation, we need to combine 512 single topologies in a single one, creating one solvent consisting of 512 molecules.[20] No worries, you do

[19] 1.4 nm is a value used very commonly in recent atomistic simulations. Its justifications comes from water simulations - at 1.4 nm, the coulombic interactions between two water molecules drop below 1% of their magnitude in the first molecule shell. For general systems possibly having much lower electrostatic shielding, this value is somewhat arbitrary, but it can be seen as a part of the model's parametrisation.

[20] When looking at the input file (already this week if you are curious, otherwise in the next exercise for sure), you will see that there is a switch to chose the number of solute and solvent molecules. While this is nicely working for

not have to redo the work you did before 512 times - for this, there is a GROMOS++ program:

```
com_top @topo 512:topo/EGM.top @param 1 @solv 1 > topo/EGM_512.top
```

`EGM.top` is the single molecule topology you just wrote, and we told the program to use it 512 times. The `@param 1` and `@solv 1` flags tell to use the parameters and solvent definitions of the first topology.[21] This writes a new topology file called `EGM_512.top`, which now contains the 512 solute molecules mentioned before. Go ahead and have a look at the file, checking the differences to the single topology you just wrote by hand.

2.3.3 Simulation Setup

As stated in Section 1, the generation of input files and starting configuration files are beyond the scope of this exercise. For this reason, they are provided to you ready-for-use. Except for the `topo` directory you have worked in until now, you will see four further directories in the main exercise directory: `crd`, `GAS`, `LIQ` and `WAT`. `crd` contains the starting configurations for the different simulations, go have a look if you are curious. The other directories contain everything else needed to run the calculations mentioned in Section 2.3.1, namely *input files* and *running scripts*. The latter are automatically generated scripts that prepare the GROMOS simulations, run the calculations, clean up the files, and, if needed, start the next job. The input files, as mentioned in the very beginning, contain all switches and parameters specifying the desired simulation run. You will look at them in more details in the next exercises. The most important characteristics of the three simulations are summarised in Table 3.

	GAS	LIQ	WAT
number of independent jobs	3	6	3
simulation time per job	100 ps	500 ps	500 ps
total simulation time	300 ps	3 ns	1.5 ns
simulation temperature	298.15 K	298.15 K	298.15 K
simulation pressure	–	1 bar	1 bar
number of solute molecules	512	512	1
number of solvent molecules	0	0	1024

Table 3: Characteristics of the Simulations

Note that there are several sequentially numbered input files and scripts in each directory. In general, we are used to subdivide longer jobs in independent, sequentially starting subjobs, connected only via the end configuration of the first job serving as an input configuration to the next. Besides a number of general advantages[22], in our case a number of independent energy trajectories will make the analysis in terms of convergence much more convenient.

---

the solvent (we will use that to have N_{wat} water molecules in our solvated simulation), this functionality has not been implemented to date for the solute, despite the switch. This is the reason for the workaround using `com_top`.

[21] For this simple example, this does not sound very relevant - but `com_top` does also allow to combine various different topologies to a single one - in which case the user has to decide which parameters shall be valid in the combined topology. Obviously, this implies that the topologies to be combined are compatible with each other - at least one topology needs to contain the parameters relevant to all molecules to be combined.

[22] Circumvention of queue time, memory and file size limitations as well as easier restarting and less loss in the event of crashes, just to name a few.

2.3.4 Submitting and Checking Your Simulations

Once that everything is ready, start your first run by navigating to the folder `LIQ` and sending the first job to the queue

```
qsub -N liq_1 -cwd -j y -o liq_1.o ./liq_1.run
```

Check that your simulation actually starts to run[23] - if your job is finished after a few second, you most probably had a crash during initialisation. In this case, go to the folder, and have a look if you can understand the reason by looking at the file `liq_1.omd`. Ask an assistant for help if necessary. If everything seems fine, repeat the above step correspondingly for the GAS and the WAT simulations in the respective folders.

The next part of doing simulations should be waiting while the computer is doing its part of the labour. Unfortunately, this is not strictly true in practise - it can feel much more like baby-sitting, especially when working with little known systems or new methods or programs. It is important to check from time to time whether the simulations are still running and the system is behaving in the desired way. As you will only learn to do analysis on the systems in the coming week, you do not have to worry about it this time - your assistants will have a backup solution for the worst case.[24] If your runs are still up a few minutes after you started them, you are done for this week!

3 Week 2 - Analysis

Last week, we created a model for ethylene glycol monoacetate (EGM) and set up a number of simulations to assess the quality of the model. This assessment is the task this week.

3.1 Status

Navigate to your simulation directory. You should find the following additional files compared to last week:

- compressed coordinate trajectories `*.trc.gz`
- compressed energy trajectories `*.tre.gz`
- GROMOS output files `*.omd`
- queue output files `*.o`

Start by checking that all simulations finished successfully. To do this, open one of the GROMOS output file. You can check the first part, everything before the lines

```
================================================
 MAIN MD LOOP
================================================
```

[23] Use the command `queue` to check the current status of the queue - see the Document "CSCBP Computational Setup" for further information.

[24] In case you anyway want to check, do not forget that from home, you need a running VPN connection to access to `beaver`. If you want to use any program having a graphic user interface, use `ssh -X` when logging in to `beaver`. Check the "CSCBP Computational Setup" document for further information.

to see how the simulation was set up. Then move to the very end of the file. You will find a summary of the simulation and, hopefully, the line

```
MD++ finished successfully
```

Check that all simulations finished successfully.[25] If you notice any problem with your simulations, try to find out what the problem could be by checking the messages in the output file. Then talk to an assistant to check how to solve the problem.

3.2 Visualisation

In a first step, we would like to visualise our simulations, thereby getting a first feeling of what is happening. Let us start with the gas-phase simulations by moving to the `GAS` directory. The coordinates of the system during the simulation run are buried within the coordinate trajectory files. In order to visualise them in an external program like `vmd`, we must extract them. The `GROMOS++` program for this task is called `frameout`. Create a file called `frameout.arg` within the `GAS` folder with the following content:

```
@topo         ../topo/EGM_512.top
@pbc          r gbond
@spec         ALL
@outformat    pdb
@include      ALL
@time         0  2
@single
@traj         gas_2.trc.gz gas_3.trc.gz
```

Please check the `GROMOS++` doxygen (navigate to `available`, then find `frameout`) for further documentation on the options chosen above. You can then call `frameout` as[26]

```
frameout @f frameout.arg
```

Then, open `vmd` on your local machine and load the file just created under `File`→`New Molecule`. It should be named `FRAME_00001.pdb`. In the gas phase, you will not see a lot initially, as the box is extremely large compared to the molecules. We should therefore choose one molecule to focus on - you can do that *via* `Graphics`→`Representations`. Under `Selected Atoms`, replace `all` by `resid 25`[27] and press `Enter` on your keyboard. Then, click on the "Display" window, and press `=` on your keyboard. Your view should now be centred on the chosen molecule. You can now play around with the different options in the "Graphical Representations" window to change the appearance of your molecule, and view different viewing angles and zooms in the "Display" window. To look at the other frames of the simulation, the camera needs to follow the chosen molecule. This is done using `Extensions`→`Analysis`→`RMSD Trajectory Tool`. In the top left corner of the new

[25] *Useful trick:* To rapidly check a larger number of files, try a command like

```
tail -n5 *.omd
```

or

```
for f in *.omd; do echo $f; grep 'MD++ finished successfully' $f; done
```

[26] Note that you could also type the arguments directly, without the use of an argument file, just as we did using `check_top`.

[27] As you might have guessed, any number between 1 and 512 (for the GAS and LIQ simulations) after `resid` will give you access to a specific molecule.

window, replace `protein` by `resid 25`, then click `ALIGN`. Go back to the "Display" window, hit `=` once more, then go to the "VMD Main" window and click the small "Play" icon in the lower right corner of the window. Feel free to play around a bit with VMD, then repeat the steps just done for the LIQ and WAT simulations.
You now have the tools to answer Question 4.2.1

3.3 Energy Trajectory Analysis

Now it is time to look at other properties of the system along the course of the simulation. A large number of observables gets calculated during the simulation run and saved in the energy trajectories. The GROMOS++ program `ene_ana` extracts values from the trajectory, saves them in time series and calculates the average, standard deviation and error[28] over the simulation run. The properties that `ene_ana` is aware of are defined in a library. You can create your own (given a certain knowledge of the structure of the energy trajectories) or modify the one standardly distributed with GROMOS. For our purposes, however, the standard library is more than enough. Create a file called `ene_ana.arg` within the folder `LIQ` with the following content:

```
@topo         ../topo/EGM_512.top
@library      /usr/local/gromos-0915/share/gromos++/ene_ana.md++.lib
@prop         densit
              totpot
@time         0  2
@en_files     liq_2.tre.gz liq_3.tre.gz liq_4.tre.gz
              liq_5.tre.gz liq_6.tre.gz
```

`densit` and `totpot` are two of these properties defined in the library. Open the one we are using here, and have a look at the properties defined. You will need at least one other property to answer the questions at the end of this exercise, and many more in the exercises to come! You can then call `ene_ana` as

```
ene_ana @f ene_ana.arg > ene_liq.out
```

As mentioned, `ene_ana` returns the average values of the chosen properties, with standard deviation and errors (check the file `ene_liq.out`). Further on, it also writes a timeseries for each property separately, see for example `totpot.dat`. A very simple way to have a quick look at this timeseries is by using `xmgrace totpot.dat`. `xmgrace` has a lot of options to customise your plot, just explore a bit! Note that `ene_ana` always chooses the same file names for the timeseries, make sure that you do not overwrite important data. Repeat the steps just done for the GAS simulations using an `ene_ana.arg` file in the `GAS` folder containing

```
@topo         ../topo/EGM_512.top
@library      /usr/local/gromos-0915/share/gromos++/ene_ana.md++.lib
@prop         totpot
@time         0  2
@en_files     gas_2.tre.gz gas_3.tre.gz
```

You now have the tools to answer Questions 4.2.2 and 4.2.3

[28] The errors are calculated using block averages of varying sizes. Thereby averages taken over different intervals of the total simulation time are compared to get an estimate of the error. More details can be found in Ref [6].

4 Report

4.1 Format

Just as for experimental approaches, mastering the technique is only one component in the scientific investigation of a given problem. Equally important components - in experiment as well as in simulation - are to:

- Formulate the question clearly
- Design an appropriate experiment to answer the question
- Interpret the results in terms of the question
- Be aware of the shortcomings and approximations of the employed method

To train these components also (at least to some extent), we expect you to hand in a **short report** after each exercise series. This report should be a bit like the "results and discussion" section of a scientific article. No need to repeat all what you did. Just quote your main results and observations, possibly using tables or/and graphs, and discuss what scientific message can be extracted from them.

To help you, at the end of each exercise series, you will find a "report" section with two subsections (here, these are the next two subsections): **simulation results** and **thinking questions**. The first one provides a hint on how you could structure the discussion of the results of the specific exercise in your report. The second one asks "outlook" questions related to the theme of the exercise (and the corresponding lecture material), which should also be answered in your report.

Try to keep you report **short** and **precise**. If possible, keep its length to a maximum[29] of **two A4 pages text** (*i.e.* excluding the space taken by possible graphs or tables). The **deadline** to hand in your report is the end of the week following second week of the exercise (see front page of this document for the exact date). Please hand in your report **to the responsible assistant** either *via* e-mail (**one single printable PDF document!**) or on paper. You can find the contact details of the assistants on the course web page.

4.2 Simulation Results

4.2.1 Visualisation

(a) Give a qualitative description of the conformational behaviour of one EGM molecule in the gas phase, in the liquid, and in water. What are the most striking differences? Can you explain the reason for these differences?

(b) Repeat this analysis considering a few different single molecules in the liquid and the gas phase. Are the conformational behaviours similar for all these molecules, *i.e.* are your observations noted in (a) representative?

(c) Direct visualisation is a very crude method to monitor the conformational behaviour of a molecule (but it is not a bad start to get a "feeling"!). Can you suggest observables we could

[29] But we are not going to hang anyone if it is three (or even four) pages, so no need to decrease the character font to 6 points so as to squeeze all on two pages!

calculate along the trajectories (*i.e.* quantities that can be given a value for each trajectory frame), so as to characterise the conformational behaviour on a quantitative basis?

4.2.2 Convergence

(a) The `ene_ana` input file given for the LIQ phase (see Section 3.3) omits the first trajectory file. Repeat the analysis including also the first trajectory file and plot the timeseries of the total potential energy and the density. What do you observe? What do you conclude concerning the starting configuration that was provided to you? When is it sensible to discard an initial piece of simulation when doing the analysis?

(b) Again omitting the first trajectory, redo the analysis using 1, 2, 3, 4 and all 5 trajectories. From the output of `ene_ana`, read out the average and the error estimate. Plot the density and the heat of vaporisation[30] (including error bars) as a function of the number of trajectories considered in the calculation. Was the total simulation length chosen long enough?

(c) The WAT simulation alone is not sufficient to calculate the hydration free energy ΔG_{wat} of EGM (you will see this in Exercise 5). But we can use it to estimate the hydration enthalpy ΔH_{wat} based on the solute-solvent non-bonded energy. Write the corresponding equation and use `ene_ana` to estimate the value based on your simulations.

(d) The approach in (c) is not rigorously correct. Why? How could we make it right?

4.2.3 Quality of the Model / Parameter Transferability

(a) Report the values of ρ and ΔH_{vap} you calculated for EGM (including an error bar estimate) and compare these with the experimental data in Table 2. Is the agreement as good as for EAE and PPL?

(b) From this comparison, is the transfer of EAE and PPL parameters to construct a model for EGM appropriate? If not, what could be the reason for the breakdown of the transferability assumption?

(c) If you wanted to refine your initial EGM model to improve agreement with experiment by slightly adjusting parameters, what kind of parameters would have the largest influence on ρ and on ΔH_{vap}?

4.3 Thinking Questions

Answer the following questions:

1. Do the atomic partial charges of your EGM molecule (Figure 3a) follow more or less what you expect based on chemical intuition, *e.g.* considering the electronegativities of the different atoms?

2. Use the `LJPARAMETERS` block of your topology file to calculate the pairwise C_6 and C_{12} Lennard-Jones interaction coefficients for the (normal) interactions between the CH2-CH2,

[30] You can keep the calculation for the GAS potential energy unchanged.

CH2-CH3 and CH3-CH3 atom-type pairs. Convert these values[31] to well depths (ϵ) and distances at the minimum (R_{min}) of the Lennard-Jones curve. Comment on the differences between the three pairs of values.

3. In the gas-phase simulations, we started with molecules being separated by at least 500 nm, simulated for 300 ps at 298.15 K, used a cutoff of 1.4 nm, and hoped that the molecules would never collide (or even just interact). Do a back-of-the-envelope calculation showing that this is a reasonable assumption. For this, you need to estimate the average translational velocity of a gas-phase EGM molecule at the selected temperature.

4. Have a look at the formula to calculate the enthalpy of vaporisation given in Equation 2 (Section 2.3.1). Why did we use the average total potential energy $\langle \mathcal{U} \rangle$ and not the average total energy $\langle \mathcal{H} \rangle = \langle \mathcal{K} + \mathcal{U} \rangle$? Why did we have to add RT?

5. When you look up for experimental data on the enthalpy of vaporisation of liquids, you typically find two types of values, both corresponding to a standard pressure of 1 bar: a value at the boiling point T_b of the liquid and a value the standard temperature of 298.15 K. What are the two different types of experimental measurements leading to these two values? In force-field parametrisation, we preferably use the standard value to be compared with a simulation at 298.15 K (as we did for EGM). Why is it so?

A Available Files

In the main exercise directory `ex1/`, you will find `ex1.pdf`, the digital version of the document you are reading, and five additional directories, namely

- `topo/`
  Contains `EGM.top`, a sketch for the topology file to be created during this exercise.

- `crd/`
  Contains the starting coordinates for the systems you will simulate to validation the topology (`EGM_gas.g96`, `EGM_liq.g96` and `EGM_h2o.g96`) as well as the coordinates of a single EGM molecule (`EGM.g96`) used in a first consistency check in Section 2.2.5.

- `GAS/`
  Three input files (`*.imd`) and three script files (`*.run`) ready to be used to run the gas phase simulation, once the topology is created.

- `LIQ/`
  Six input files (`*.imd`) and six script files (`*.run`) ready to be used to run the liquid phase simulation, once the topology is created.

- `WAT/`
  Three input files (`*.imd`) and three script files (`*.run`) ready to be used to run the solvated simulation, once the topology is created.

[31] The equations are in the lecture notes

References

[1] Horta, B.A.C.; Fuchs, P.F.J.; van Gunsteren, W.F.; Hunenberger, P.H.; *J. Chem. Theory Comput.* **2011**, 7, 10161031.

[2] Berendsen, H.J.C.; Postma, J.P.M.; van Gunsteren, W.F.; Hermans, J.; In *Intermolecular Forces*, edited by B. Pullman; Reidel, Dordrecht, 1981, p. 331.

[3] http://www.hbcpnetbase.com

[4] Riddick, J.A.; Bunger, W.B.; Sakano,T.K.; *Organic solvents,physical properties and methods of purification*; John Wiley & Sons, New York, 1986.

[5] Acree Jr, W. & Chickos, J.S.; *J. Phys. Chem. Ref. Data*, **2010**, 39, 043101.

[6] Allen, M.P.; and Tildesley, D.J.; *Computer Simulation of Liquids*; Clarendon Press, Oxford, 1987.