5.7 File formats

5.7.1 Topology file

The topology file is built following the GROMACS specification for a molecular topology. A `*.top` file can be generated by `pdb2gmx`. All possible entries in the topology file are listed in Tables 5.4 and 5.5. Also tabulated are: all the units of the parameters, which interactions can be perturbed for free energy calculations, which bonded interactions are used by `grompp` for generating exclusions, and which bonded interactions can be converted to constraints by `grompp`.

Parameters

interaction type	directive	# at.	f. tp	parameters	F. E.
mandatory	`defaults`			non-bonded function type; combination rule$^{(cr)}$; generate pairs (no/yes); fudge LJ (); fudge QQ ()	
mandatory	`atomtypes`			atom type; m (u); q (e); particle type; V$^{(cr)}$; W$^{(cr)}$	
	`bondtypes`	(see Table 5.5, directive `bonds`)			
	`pairtypes`	(see Table 5.5, directive `pairs`)			
	`angletypes`	(see Table 5.5, directive `angles`)			
	`dihedraltypes`$^{(*)}$	(see Table 5.5, directive `dihedrals`)			
	`constrainttypes`	(see Table 5.5, directive `constraints`)			
LJ	`nonbond_params`	2	1	$V^{(cr)}$; $W^{(cr)}$	
Buckingham	`nonbond_params`	2	2	a (kJ mol^{-1}); b (nm^{-1}); c_6 (kJ mol^{-1} nm^6)	

Molecule definition(s)

mandatory	`moleculetype`			molecule name; $n_{ex}^{(nrexcl)}$	
mandatory	`atoms`	1		atom type; residue number; residue name; atom name; charge group number; q (e); m (u)	type q, m
intra-molecular interaction and geometry definitions as described in Table 5.5					

System

mandatory	`system`	system name
mandatory	`molecules`	molecule name; number of molecules

'# at' is the required number of atom type indices for this directive
'f. tp' is the value used to select this function type
'F. E.' indicates which of the parameters for this interaction can be
interpolated during free energy calculations
$^{(cr)}$ the combination rule determines the type of LJ parameters, see 5.3.2
$^{(*)}$ for `dihedraltypes` one can specify 4 atoms or the inner (outer for improper) 2 atoms
$^{(nrexcl)}$ exclude neighbors n_{ex} bonds away for non-bonded interactions
For free energy calculations, type, q and m or no parameters should be added
for topology 'B' ($\lambda = 1$) on the same line, after the normal parameters.

Table 5.4: The topology (`*.top`) file.

Table 5.5: Details of `[ moleculetype ]` directives

Name of interaction	Topology file directive	num. atoms[*]	func. type[†]	Order of parameters and their units	use in F.E.?[‡]	Cross-references
bond	`bonds`[§,¶]	2	1	b_0 (nm); k_b (kJ mol^{-1} nm^{-2})	all	4.2.1
G96 bond	`bonds`[§,¶]	2	2	b_0 (nm); k_b (kJ mol^{-1} nm^{-4})	all	4.2.1
Morse	`bonds`[§,¶]	2	3	b_0 (nm); D (kJ mol^{-1}); β (nm^{-1})	all	4.2.2
cubic bond	`bonds`[§,¶]	2	4	b_0 (nm); $C_{i=2,3}$ (kJ mol^{-1} nm^{-i})		4.2.3
connection	`bonds`[§]	2	5			5.4
harmonic potential	`bonds`	2	6	b_0 (nm); k_b (kJ mol^{-1} nm^{-2})	all	4.2.1,5.4
FENE bond	`bonds`[§]	2	7	b_m (nm); k_b (kJ mol^{-1} nm^{-2})		4.2.4
tabulated bond	`bonds`[§]	2	8	table number (≥ 0); k (kJ mol^{-1})	k	4.2.14
tabulated bond[‖]	`bonds`	2	9	table number (≥ 0); k (kJ mol^{-1})	k	4.2.14,5.4
restraint potential	`bonds`	2	10	low, up$_1$, up$_2$ (nm); k_{dr} (kJ mol^{-1} nm^{-2})	all	4.3.5
extra LJ or Coulomb	`pairs`	2	1	V^{**}; W^{**}	all	5.3.4
extra LJ or Coulomb	`pairs`	2	2	fudge QQ (); q_i, q_j (e), V^{**}; W^{**}		5.3.4
extra LJ or Coulomb	`pairs_nb`	2	1	q_i, q_j (e); V^{**}; W^{**}		5.3.4
angle	`angles`[¶]	3	1	θ_0 (deg); k_θ (kJ mol^{-1} rad^{-2})	all	4.2.5
G96 angle	`angles`[¶]	3	2	θ_0 (deg); k_θ (kJ mol^{-1})	all	4.2.6
cross bond-bond	`angles`	3	3	r_{1e}, r_{2e} (nm); $k_{rr'}$ (kJ mol^{-1} nm^{-2})		4.2.9
cross bond-angle	`angles`	3	4	r_{1e}, r_{2e} r_{3e} (nm); $k_{r\theta}$ (kJ mol^{-1} nm^{-2})		4.2.10
Urey-Bradley	`angles`[¶]	3	5	θ_0 (deg); k_θ (kJ mol^{-1} rad^{-2}); r_{13} (nm); k_{UB} (kJ mol^{-1} nm^{-2})	all	4.2.8
quartic angle	`angles`[¶]	3	6	θ_0 (deg); $C_{i=0,1,2,3,4}$ (kJ mol^{-1} rad^{-i})		4.2.11

[*]The required number of atom indices for this directive
[†]The index to use to select this function type
[‡]Indicates which of the parameters for this interaction can be interpolated during free energy calculations
[§]This interaction type will be used by `grompp` for generating exclusions
[¶]This interaction type can be converted to constraints by `grompp`
[**]The combination rule determines the type of LJ parameters, see 5.3.2
[‖]No connection, and so no exclusions, are generated for this interaction

Table 5.5: Details of `[ moleculetype ]` directives

Name of interaction	Topology file directive	num. atoms[*]	func. type[†]	Order of parameters and their units	use in F.E.?[‡]	Cross-references
tabulated angle	`angles`	3	8	table number ($\geq$ 0); k (kJ mol^{-1})	k	4.2.14
restricted bending potential	`angles`	3	10	θ_0 (deg); k_θ (kJ mol^{-1})		4.2.7
proper dihedral	`dihedrals`	4	1	ϕ_s (deg); k_ϕ (kJ mol^{-1}); multiplicity	ϕ, k	4.2.13
improper dihedral	`dihedrals`	4	2	ξ_0 (deg); k_ξ (kJ mol^{-1} rad^{-2})	all	4.2.12
Ryckaert-Bellemans dihedral	`dihedrals`	4	3	C_0, C_1, C_2, C_3, C_4, C_5 (kJ mol^{-1})	all	4.2.13
periodic improper dihedral	`dihedrals`	4	4	ϕ_s (deg); k_ϕ (kJ mol^{-1}); multiplicity	ϕ, k	4.2.12
Fourier dihedral	`dihedrals`	4	5	C_1, C_2, C_3, C_4 (kJ mol^{-1})	all	4.2.13
tabulated dihedral	`dihedrals`	4	8	table number ($\geq$ 0); k (kJ mol^{-1})	k	4.2.14
proper dihedral (multiple)	`dihedrals`	4	9	ϕ_s (deg); k_ϕ (kJ mol^{-1}); multiplicity	ϕ, k	4.2.13
restricted dihedral	`dihedrals`	4	11	ϕ_0 (deg); k_ϕ (kJ mol^{-1})		4.2.13
combined bending-torsion potential	`dihedrals`	4	10	a_0, a_1, a_2, a_3, a_4 (kJ mol^{-1})		4.2.13
exclusions	`exclusions`	1		one or more atom indices		5.4
constraint	`constraints`[§]	2	1	b_0 (nm)	all	4.5,5.5
constraint[‖]	`constraints`	2	2	b_0 (nm)	all	4.5,5.5,5.4
SETTLE	`settles`	1	1	$d_{\rm OH}$, $d_{\rm HH}$ (nm)		3.6.1,5.5
2-body virtual site	`virtual_sites2`	3	1	a ()		4.7
3-body virtual site	`virtual_sites3`	4	1	a, b ()		4.7
3-body virtual site (fd)	`virtual_sites3`	4	2	a (); d (nm)		4.7
3-body virtual site (fad)	`virtual_sites3`	4	3	θ (deg); d (nm)		4.7
3-body virtual site (out)	`virtual_sites3`	4	4	a, b (); c (nm^{-1})		4.7
4-body virtual site (fdn)	`virtual_sites4`	5	2	a, b (); c (nm)		4.7
N-body virtual site (COG)	`virtual_sitesn`	1	1	one or more constructing atom indices		4.7
N-body virtual site (COM)	`virtual_sitesn`	1	2	one or more constructing atom indices		4.7
N-body virtual site (COW)	`virtual_sitesn`	1	3	one or more pairs consisting of constructing atom index and weight		4.7
position restraint	`position_restraints`	1	1	k_x, k_y, k_z (kJ mol^{-1} nm^{-2})	all	4.3.1

Table 5.5: Details of `[ moleculetype ]` directives

Name of interaction	Topology file directive	num. atoms*	func. type†	Order of parameters and their units	use in F.E.?‡	Cross-references
flat-bottomed position restraint	`position_restraints`	1	2	g, r (nm), k (kJ mol^{-1} nm^{-2})		4.3.2
distance restraint	`distance_restraints`	2	1	type; label; low, up$_1$, up$_2$ (nm); weight ()		4.3.5
dihedral restraint	`dihedral_restraints`	4	1	ϕ_0 (deg); $\Delta\phi$ (deg);	all	4.3.4
orientation restraint	`orientation_restraints`	2	1	exp.; label; α; c (U nm$^{\alpha}$); obs. (U); weight (U^{-1})		4.3.6
angle restraint	`angle_restraints`	4	1	θ_0 (deg); k_c (kJ mol^{-1}); multiplicity	θ, k	4.3.3
angle restraint (z)	`angle_restraints_z`	2	1	θ_0 (deg); k_c (kJ mol^{-1}); multiplicity	θ, k	4.3.3

Description of the file layout:

- Semicolon (;) and newline characters surround comments
- On a line ending with \ the newline character is ignored.
- Directives are surrounded by `[` and `]`
- The topology hierarchy (which must be followed) consists of three levels:
  - the parameter level, which defines certain force-field specifications (see Table 5.4)
  - the molecule level, which should contain one or more molecule definitions (see Table 5.5)
  - the system level, containing only system-specific information (`[ system ]` and `[ molecules ]`)
- Items should be separated by spaces or tabs, not commas
- Atoms in molecules should be numbered consecutively starting at 1
- Atoms in the same charge group must be listed consecutively
- The file is parsed only once, which implies that no forward references can be treated: items must be defined before they can be used
- Exclusions can be generated from the bonds or overridden manually
- The bonded force types can be generated from the atom types or overridden per bond
- It is possible to apply multiple bonded interactions of the same type on the same atoms
- Descriptive comment lines and empty lines are highly recommended
- Starting with GROMACS version 3.1.3, all directives at the parameter level can be used multiple times and there are no restrictions on the order, except that an atom type needs to be defined before it can be used in other parameter definitions
- If parameters for a certain interaction are defined multiple times for the same combination of atom types the last definition is used; starting with GROMACS version 3.1.3 `grompp` generates a warning for parameter redefinitions with different values
- Using one of the `[ atoms ]`, `[ bonds ]`, `[ pairs ]`, `[ angles ]`, etc. without having used `[ moleculetype ]` before is meaningless and generates a warning
- Using `[ molecules ]` without having used `[ system ]` before is meaningless and generates a warning.
- After `[ system ]` the only allowed directive is `[ molecules ]`
- Using an unknown string in `[ ]` causes all the data until the next directive to be ignored and generates a warning

Here is an example of a topology file, `urea.top`:

```
;
;       Example topology file
;
; The force-field files to be included
#include "amber99.ff/forcefield.itp"

[ moleculetype ]
; name  nrexcl
Urea         3

[ atoms ]
   1  C  1  URE      C      1     0.880229  12.01000   ; amber C  type
   2  O  1  URE      O      2    -0.613359  16.00000   ; amber O  type
   3  N  1  URE     N1      3    -0.923545  14.01000   ; amber N  type
   4  H  1  URE    H11      4     0.395055   1.00800   ; amber H  type
   5  H  1  URE    H12      5     0.395055   1.00800   ; amber H  type
   6  N  1  URE     N2      6    -0.923545  14.01000   ; amber N  type
   7  H  1  URE    H21      7     0.395055   1.00800   ; amber H  type
   8  H  1  URE    H22      8     0.395055   1.00800   ; amber H  type

[ bonds ]
    1 2
    1 3
    1   6
    3 4
    3 5
    6 7
    6 8

[ dihedrals ]
;   ai    aj    ak    al  funct  definition
     2     1     3     4   9
     2     1     3     5   9
     2     1     6     7   9
     2     1     6     8   9
     3     1     6     7   9
     3     1     6     8   9
     6     1     3     4   9
     6     1     3     5   9

[ dihedrals ]
     3     6     1     2   4
     1     4     3     5   4
     1     7     6     8   4

[ position_restraints ]
; you wouldn't normally use this for a molecule like Urea,
; but we include it here for didactic purposes
; ai   funct    fc
   1     1     1000    1000    1000 ; Restrain to a point
   2     1     1000       0    1000 ; Restrain to a line (Y-axis)
   3     1     1000       0       0 ; Restrain to a plane (Y-Z-plane)

[ dihedral_restraints ]
; ai   aj    ak    al  type  label  phi  dphi  kfac  power
    3    6     1     2     1      1  180     0     1      2
    1    4     3     5     1      1  180     0     1      2

; Include TIP3P water topology
#include "amber99/tip3p.itp"

[ system ]
Urea in Water

[ molecules ]
;molecule name   nr.
Urea             1
SOL              1000
```

Here follows the explanatory text.

`#include "amber99.ff/forcefield.itp"` **:** this includes the information for the force field you are using, including bonded and non-bonded parameters. This example uses the AMBER99 force field, but your simulation may use a different force field. `grompp` will automatically go and find this file and copy-and-paste its content. That content can be seen in `share/top/amber99.ff/forcefield.itp`, and it is

```
#define _FF_AMBER
#define _FF_AMBER99

[ defaults ]
; nbfunc        comb-rule       gen-pairs       fudgeLJ fudgeQQ
1               2               yes             0.5     0.8333

#include "ffnonbonded.itp"
#include "ffbonded.itp"
#include "gbsa.itp"
```

The two `#define` statements set up the conditions so that future parts of the topology can know that the AMBER 99 force field is in use.

`[ defaults ]` **:**

- `nbfunc` is the non-bonded function type. Use 1 (Lennard-Jones) or 2 (Buckingham)
- `comb-rule` is the number of the combination rule (see 5.3.2).
- `gen-pairs` is for pair generation. The default is ‘no’, *i.e.* get 1-4 parameters from the pairtypes list. When parameters are not present in the list, stop with a fatal error. Setting ‘yes’ generates 1-4 parameters that are not present in the pair list from normal Lennard-Jones parameters using `fudgeLJ`

- `fudgeLJ` is the factor by which to multiply Lennard-Jones 1-4 interactions, default 1
- `fudgeQQ` is the factor by which to multiply electrostatic 1-4 interactions, default 1
- N is the power for the repulsion term in a 6-N potential (with nonbonded-type Lennard-Jones only), starting with GROMACS version 4.5, `mdrun` also reads and applies N, for values not equal to 12 tabulated interaction functions are used (in older version you would have to use user tabulated interactions).

Note that `gen-pairs`, `fudgeLJ`, `fudgeQQ`, and N are optional. `fudgeLJ` is only used when generate pairs is set to 'yes', and `fudgeQQ` is always used. However, if you want to specify N you need to give a value for the other parameters as well.

Then some other `#include` statements add in the large amount of data needed to describe the rest of the force field. We will skip these and return to `urea.top`. There we will see

`[ moleculetype ]` **:** defines the name of your molecule in this `*.top` and nrexcl = 3 stands for excluding non-bonded interactions between atoms that are no further than 3 bonds away.

`[ atoms ]` **:** defines the molecule, where `nr` and `type` are fixed, the rest is user defined. So `atom` can be named as you like, `cgnr` made larger or smaller (if possible, the total charge of a charge group should be zero), and charges can be changed here too.

`[ bonds ]` **:** no comment.

`[ pairs ]` **:** LJ and Coulomb 1-4 interactions

`[ angles ]` **:** no comment

`[ dihedrals ]` **:** in this case there are 9 proper dihedrals (funct = 1), 3 improper (funct = 4) and no Ryckaert-Bellemans type dihedrals. If you want to include Ryckaert-Bellemans type dihedrals in a topology, do the following (in case of *e.g.* decane):

```
[ dihedrals ]
;  ai    aj    ak    al  funct        c0        c1        c2
    1     2     3     4     3
    2     3     4     5     3
```

In the original implementation of the potential for alkanes [128] no 1-4 interactions were used, which means that in order to implement that particular force field you need to remove the 1-4 interactions from the `[ pairs ]` section of your topology. In most modern force fields, like OPLS/AA or Amber the rules are different, and the Ryckaert-Bellemans potential is used as a cosine series in combination with 1-4 interactions.

`[ position_restraints ]` **:** harmonically restrain the selected particles to reference positions (4.3.1). The reference positions are read from a separate coordinate file by `grompp`.

`[ dihedral_restraints ]` **:** restrain selected dihedrals to a reference value. The implementation of dihedral restraints is described in section 4.3.4 of the manual. The parameters specified in the [dihedral_restraints] directive are as follows:

- `type` has only one possible value which is 1
- `label` is unused and has been removed from the code.

- phi is the value of ϕ_0 in eqn. 4.84 and eqn. 4.85 of the manual.
- dphi is the value of $\Delta\phi$ in eqn. 4.85 of the manual.
- kfac is analogous to fac in the implementation of distance restraints. It is the factor by which the force constant is multiplied. By doing so, different restraints can be maintained with different force constants.
- power is unused and has been removed from the code.

#include "tip3p.itp" **:** includes a topology file that was already constructed (see section 5.7.2).

[system] **:** title of your system, user-defined

[molecules] **:** this defines the total number of (sub)molecules in your system that are defined in this *.top. In this example file, it stands for 1 urea molecule dissolved in 1000 water molecules. The molecule type SOL is defined in the tip3p.itp file. Each name here must correspond to a name given with [moleculetype] earlier in the topology. The order of the blocks of molecule types and the numbers of such molecules must match the coordinate file that accompanies the topology when supplied to grompp. The blocks of molecules do not need to be contiguous, but some tools (e.g. genion) may act only on the first or last such block of a particular molecule type. Also, these blocks have nothing to do with the definition of groups (see sec. 3.3 and sec. 8.1).

5.7.2 Molecule.itp file

If you construct a topology file you will use frequently (like the water molecule, tip3p.itp, which is already constructed for you) it is good to make a molecule.itp file. This only lists the information of one particular molecule and allows you to re-use the [moleculetype] in multiple systems without re-invoking pdb2gmx or manually copying and pasting. An example urea.itp follows:

```
[ moleculetype ]
; molname nrexcl
URE 3

[ atoms ]
   1   C   1   URE      C      1     0.880229  12.01000   ; amber C  type
...
   8   H   1   URE    H22      8     0.395055   1.00800   ; amber H  type

[ bonds ]
    1  2
...
    6  8
[ dihedrals ]
;    ai     aj     ak     al  funct   definition
      2      1      3      4    9
...
```

```
     6     1     3     5    9
[ dihedrals ]
     3     6     1     2    4
     1     4     3     5    4
     1     7     6     8    4
```

Using `*.itp` files results in a very short `*.top` file:

```
;
;       Example topology file
;
; The force field files to be included
#include "amber99.ff/forcefield.itp"

#include "urea.itp"

; Include TIP3P water topology
#include "amber99/tip3p.itp"

[ system ]
Urea in Water

[ molecules ]
;molecule name   nr.
Urea             1
SOL              1000
```

5.7.3 Ifdef statements

A very powerful feature in GROMACS is the use of `#ifdef` statements in your `*.top` file. By making use of this statement, and associated `#define` statements like were seen in `amber99.ff/forcefield.itp` earlier, different parameters for one molecule can be used in the same `*.top` file. An example is given for TFE, where there is an option to use different charges on the atoms: charges derived by De Loof *et al.* [129] or by Van Buuren and Berendsen [130]. In fact, you can use much of the functionality of the C preprocessor, `cpp`, because `grompp` contains similar pre-processing functions to scan the file. The way to make use of the `#ifdef` option is as follows:

- either use the option `define = -DDeLoof` in the `*.mdp` file (containing `grompp` input parameters), or use the line `#define DeLoof` early in your `*.top` or `*.itp` file; and
- put the `#ifdef` statements in your `*.top`, as shown below:

```
...


[ atoms ]
; nr     type      resnr     residu      atom       cgnr       charge          mass
```

```
#ifdef DeLoof
; Use Charges from DeLoof
   1        C        1        TFE        C        1        0.74
   2        F        1        TFE        F        1       -0.25
   3        F        1        TFE        F        1       -0.25
   4        F        1        TFE        F        1       -0.25
   5      CH2        1        TFE      CH2        1        0.25
   6       OA        1        TFE       OA        1       -0.65
   7       HO        1        TFE       HO        1        0.41
#else
; Use Charges from VanBuuren
   1        C        1        TFE        C        1        0.59
   2        F        1        TFE        F        1       -0.2
   3        F        1        TFE        F        1       -0.2
   4        F        1        TFE        F        1       -0.2
   5      CH2        1        TFE      CH2        1        0.26
   6       OA        1        TFE       OA        1       -0.55
   7       HO        1        TFE       HO        1        0.3
#endif

[ bonds ]
;  ai    aj funct           c0           c1
    6     7     1 1.000000e-01 3.138000e+05
    1     2     1 1.360000e-01 4.184000e+05
    1     3     1 1.360000e-01 4.184000e+05
    1     4     1 1.360000e-01 4.184000e+05
    1     5     1 1.530000e-01 3.347000e+05
    5     6     1 1.430000e-01 3.347000e+05
...
```

This mechanism is used by `pdb2gmx` to implement optional position restraints (4.3.1) by `#include`-ing an `.itp` file whose contents will be meaningful only if a particular `#define` is set (and spelled correctly!)

5.7.4 Topologies for free energy calculations

Free energy differences between two systems, A and B, can be calculated as described in sec. 3.12. Systems A and B are described by topologies consisting of the same number of molecules with the same number of atoms. Masses and non-bonded interactions can be perturbed by adding B parameters under the `[ atoms ]` directive. Bonded interactions can be perturbed by adding B parameters to the bonded types or the bonded interactions. The parameters that can be perturbed are listed in Tables 5.4 and 5.5. The λ-dependence of the interactions is described in section sec. 4.5. The bonded parameters that are used (on the line of the bonded interaction definition, or the ones looked up on atom types in the bonded type lists) is explained in Table 5.6. In most cases, things should work intuitively. When the A and B atom types in a bonded interaction are not all identical and parameters are not present for the B-state, either on the line or in the bonded types, `grompp` uses the A-state parameters and issues a warning. For free energy calculations, all or no parameters for topology B ($\lambda = 1$) should be added on the same line, after the normal parameters, in the same order as the normal parameters. From GROMACS 4.6 onward, if λ is treated as a

B-state atom types all identical to A-state atom types	parameters on line		parameters in bonded types				message
			A atom types		B atom types		
	A	B	A	B	A	B	
yes	+AB	−	x	x			
	+A	+B	x	x			
	−	−	−	−			error
	−	−	+AB	−			
	−	−	+A	+B			
no	+AB	−	x	x	x	x	warning
	+A	+B	x	x	x	x	
	−	−	−	−	x	x	error
	−	−	+AB	−	−	−	warning
	−	−	+A	+B	−	−	warning
	−	−	+A	x	+B	−	
	−	−	+A	x	+	+B	

Table 5.6: The bonded parameters that are used for free energy topologies, on the line of the bonded interaction definition or looked up in the bond types section based on atom types. A and B indicate the parameters used for state A and B respectively, + and − indicate the (non-)presence of parameters in the topology, x indicates that the presence has no influence.

vector, then the `bonded-lambdas` component controls all bonded terms that are not explicitly labeled as restraints. Restrain terms are controlled by the `restraint-lambdas` component.

Below is an example of a topology which changes from 200 propanols to 200 pentanes using the GROMOS-96 force field.

```
; Include force field parameters
#include "gromos43a1.ff/forcefield.itp"

[ moleculetype ]
; Name            nrexcl
PropPent          3

[ atoms ]
; nr type resnr residue atom cgnr  charge    mass  typeB chargeB  massB
  1    H    1     PROP    PH    1   0.398    1.008  CH3      0.0  15.035
  2   OA    1     PROP    PO    1  -0.548  15.9994  CH2      0.0  14.027
  3  CH2    1     PROP   PC1    1   0.150   14.027  CH2      0.0  14.027
  4  CH2    1     PROP   PC2    2   0.000   14.027
  5  CH3    1     PROP   PC3    2   0.000   15.035

[ bonds ]
;  ai    aj funct    par_A  par_B
    1     2     2    gb_1   gb_26
    2     3     2    gb_17  gb_26
    3     4     2    gb_26  gb_26
    4     5     2    gb_26

[ pairs ]
;  ai    aj funct
    1     4     1
    2     5     1

[ angles ]
;  ai    aj    ak funct     par_A   par_B
    1     2     3     2     ga_11   ga_14
    2     3     4     2     ga_14   ga_14
    3     4     5     2     ga_14   ga_14

[ dihedrals ]
;  ai    aj    ak    al funct     par_A   par_B
    1     2     3     4     1     gd_12   gd_17
    2     3     4     5     1     gd_17   gd_17

[ system ]
; Name
Propanol to Pentane

[ molecules ]
; Compound        #mols
PropPent          200
```

Atoms that are not perturbed, `PC2` and `PC3`, do not need B-state parameter specifications, since the B parameters will be copied from the A parameters. Bonded interactions between atoms that are not perturbed do not need B parameter specifications, as is the case for the last bond in the example topology. Topologies using the OPLS/AA force field need no bonded parameters at all, since both the A and B parameters are determined by the atom types. Non-bonded interactions involving one or two perturbed atoms use the free-energy perturbation functional forms. Non-bonded interactions between two non-perturbed atoms use the normal functional forms. This means that when, for instance, only the charge of a particle is perturbed, its Lennard-Jones interactions will also be affected when lambda is not equal to zero or one.

Note that this topology uses the GROMOS-96 force field, in which the bonded interactions are not determined by the atom types. The bonded interaction strings are converted by the C-preprocessor. The force-field parameter files contain lines like:

```
#define gb_26       0.1530  7.1500e+06

#define gd_17     0.000       5.86          3
```

5.7.5 Constraint forces

The constraint force between two atoms in one molecule can be calculated with the free energy perturbation code by adding a constraint between the two atoms, with a different length in the A and B topology. When the B length is 1 nm longer than the A length and lambda is kept constant at zero, the derivative of the Hamiltonian with respect to lambda is the constraint force. For constraints between molecules, the pull code can be used, see sec. 6.4. Below is an example for calculating the constraint force at 0.7 nm between two methanes in water, by combining the two methanes into one “molecule.” **Note** that the definition of a “molecule” in GROMACS does not necessarily correspond to the chemical definition of a molecule. In GROMACS, a “molecule” can be defined as any group of atoms that one wishes to consider simultaneously. The added constraint is of function type 2, which means that it is not used for generating exclusions (see sec. 5.4). Note that the constraint free energy term is included in the derivative term, and is specifically included in the `bonded-lambdas` component. However, the free energy for changing constraints is *not* included in the potential energy differences used for BAR and MBAR, as this requires reevaluating the energy at each of the constraint components. This functionality is planned for later versions.

```
; Include force-field parameters
#include "gromos43a1.ff/forcefield.itp"

[ moleculetype ]
; Name              nrexcl
Methanes               1

[ atoms ]
; nr   type   resnr  residu   atom    cgnr      charge    mass
   1    CH4     1     CH4      C1       1          0    16.043
   2    CH4     1     CH4      C2       2          0    16.043
[ constraints ]
;  ai     aj funct   length_A  length_B
    1      2     2        0.7       1.7

#include "gromos43a1.ff/spc.itp"

[ system ]
; Name
Methanes in Water

[ molecules ]
; Compound          #mols
Methanes              1
SOL                2002
```

5.7.6 Coordinate file

Files with the `.gro` file extension contain a molecular structure in GROMOS-87 format. A sample piece is included below:

```
MD of 2 waters, reformat step, PA aug-91
    6
    1WATER  OW1    1   0.126   1.624   1.679  0.1227 -0.0580  0.0434
    1WATER  HW2    2   0.190   1.661   1.747  0.8085  0.3191 -0.7791
    1WATER  HW3    3   0.177   1.568   1.613 -0.9045 -2.6469  1.3180
    2WATER  OW1    4   1.275   0.053   0.622  0.2519  0.3140 -0.1734
    2WATER  HW2    5   1.337   0.002   0.680 -1.0641 -1.1349  0.0257
    2WATER  HW3    6   1.326   0.120   0.568  1.9427 -0.8216 -0.0244
   1.82060   1.82060   1.82060
```

This format is fixed, *i.e.* all columns are in a fixed position. If you want to read such a file in your own program without using the GROMACS libraries you can use the following formats:

C-format: `"%5i%5s%5s%5i%8.3f%8.3f%8.3f%8.4f%8.4f%8.4f"`

Or to be more precise, with title *etc.* it looks like this:

```
  "%s\n", Title
  "%5d\n", natoms
  for (i=0; (i<natoms); i++) {
    "%5d%-5s%5s%5d%8.3f%8.3f%8.3f%8.4f%8.4f%8.4f\n",
      residuenr,residuename,atomname,atomnr,x,y,z,vx,vy,vz
  }
  "%10.5f%10.5f%10.5f%10.5f%10.5f%10.5f%10.5f%10.5f%10.5f\n",
    box[X][X],box[Y][Y],box[Z][Z],
    box[X][Y],box[X][Z],box[Y][X],box[Y][Z],box[Z][X],box[Z][Y]
```

Fortran format: `(i5,2a5,i5,3f8.3,3f8.4)`

So `confin.gro` is the GROMACS coordinate file and is almost the same as the GROMOS-87 file (for GROMOS users: when used with `ntx=7`). The only difference is the box for which GROMACS uses a tensor, not a vector.

5.8 Force field organization

5.8.1 Force-field files

Many force fields are available by default. Force fields are detected by the presence of `<name>.ff` directories in the `$GMXLIB/share/gromacs/top` sub-directory and/or the working directory. The information regarding the location of the force field files is printed by `pdb2gmx` so you can easily keep track of which version of a force field is being called, in case you have made modifications in one location or another. The force fields included with GROMACS are:

- AMBER03 protein, nucleic AMBER94 (Duan et al., J. Comp. Chem. 24, 1999-2012, 2003)
- AMBER94 force field (Cornell et al., JACS 117, 5179-5197, 1995)
- AMBER96 protein, nucleic AMBER94 (Kollman et al., Acc. Chem. Res. 29, 461-469, 1996)
- AMBER99 protein, nucleic AMBER94 (Wang et al., J. Comp. Chem. 21, 1049-1074, 2000)
- AMBER99SB protein, nucleic AMBER94 (Hornak et al., Proteins 65, 712-725, 2006)
- AMBER99SB-ILDN protein, nucleic AMBER94 (Lindorff-Larsen et al., Proteins 78, 1950-58, 2010)

- AMBERGS force field (Garcia & Sanbonmatsu, PNAS 99, 2782-2787, 2002)
- CHARMM27 all-atom force field (CHARM22 plus CMAP for proteins)
- GROMOS96 43a1 force field
- GROMOS96 43a2 force field (improved alkane dihedrals)
- GROMOS96 45a3 force field (Schuler JCC 2001 22 1205)
- GROMOS96 53a5 force field (JCC 2004 vol 25 pag 1656)
- GROMOS96 53a6 force field (JCC 2004 vol 25 pag 1656)
- GROMOS96 54a7 force field (Eur. Biophys. J. (2011), 40,, 843-856, DOI: 10.1007/s00249-011-0700-9)
- OPLS-AA/L all-atom force field (2001 aminoacid dihedrals)

A force field is included at the beginning of a topology file with an `#include` statement followed by `<name>.ff/forcefield.itp`. This statement includes the force-field file, which, in turn, may include other force-field files. All the force fields are organized in the same way. An example of the `amber99.ff/forcefield.itp` was shown in 5.7.1.

For each force field, there several files which are only used by `pdb2gmx`. These are: residue databases (`.rtp`, see 5.6.1) the hydrogen database (`.hdb`, see 5.6.4), two termini databases (`.n.tdb` and `.c.tdb`, see 5.6.5) and the atom type database (`.atp`, see 5.2.1), which contains only the masses. Other optional files are described in sec. 5.6.

5.8.2 Changing force-field parameters

If one wants to change the parameters of few bonded interactions in a molecule, this is most easily accomplished by typing the parameters behind the definition of the bonded interaction directly in the `*.top` file under the `[ moleculetype ]` section (see 5.7.1 for the format and units). If one wants to change the parameters for all instances of a certain interaction one can change them in the force-field file or add a new `[ ???types ]` section after including the force field. When parameters for a certain interaction are defined multiple times, the last definition is used. As of GROMACS version 3.1.3, a warning is generated when parameters are redefined with a different value. Changing the Lennard-Jones parameters of an atom type is not recommended, because in the GROMOS force fields the Lennard-Jones parameters for several combinations of atom types are not generated according to the standard combination rules. Such combinations (and possibly others that do follow the combination rules) are defined in the `[ nonbond_params ]` section, and changing the Lennard-Jones parameters of an atom type has no effect on these combinations.

5.8.3 Adding atom types

As of GROMACS version 3.1.3, atom types can be added in an extra `[ atomtypes ]` section after the the inclusion of the normal force field. After the definition of the new atom type(s), additional non-bonded and pair parameters can be defined. In pre-3.1.3 versions of GROMACS, the new atom types needed to be added in the `[ atomtypes ]` section of the force-field files, because all non-bonded parameters above the last `[ atomtypes ]` section would be overwritten using the standard combination rules.