

“要在自己的代码里找出一个bug是十分困难的。而当你认为你的代码没有错误时，那就更难了。—— Steve McConnell《代码大全》”

速度不给力? 试试国内网站 <http://t066v5.coding-pages.com>

评论与问题? 请至论坛留言 <https://Jerkwin.herokuapp.com>

GROMACS中文手册：第五章 拓扑文件

类别: 科 标签: gmx 阅读次数: 607479 版权: (CC) BY-NC-SA

5.1 简介

GROMACS运行时需要知道哪些原子及其组合对势能函数有贡献(参见第四章). 此外, 它还需要知道对于不同函数必需的参数. 所有这些都利用 **拓扑文件** `*.top` 进行描述, 它列出了每个原子的 **固定属性**. 原子类型比元素种类多得多, 但力场只对存在于生物系统中的原子类型, 外加一些金属, 离子和硅进行了参数化. 键合相互作用和一些特殊的相互作用由拓扑文件中的固定配对所决定, 必须排除某些非键相互作用(第一和第二近邻), 因为它们已经在键合相互作用中处理过了. 另外, 原子还具有 **动态属性**, 如位置, 速度和力. 严格地说, 这些不属于分子拓扑, 而是存储在坐标文件 `*.gro` (位置和速度)或轨迹文件 `*.trr` (位置, 速度, 力).

本章将描述拓扑文件, `*.top` 文件和数据库文件的设置: 各个参数代表什么, 如果需要如何/到哪里修改它们. 首先, 会解释了所有的文件格式. 然后, 在5.8.1小节说明在每个力场的文件组织形式.

注意: 如果你创建了自己的拓扑文件, 我们诚挚地希望你能够将其上传到位于 www.gromacs.org 的拓扑文件存档. 想一想, 如果在开始计算之前, 那里就已经存有你需要的拓扑文件了, 你肯定会感谢那些贡献的人. 如果你创建了新的力场或者对标准力场进行了修改, 我们同样希望你能够将其上传到力场存档.

5.2 粒子类型

在GROMACS中, 共有三种类型的粒子, 参见表 5.1. GROMACS只使用了常规原子和虚拟相互作用位点, 壳层只对极化模型, 如壳层水模型, 才是需要的[43].

表 5.1: GROMACS的粒子类型

粒子	符号
原子	A
壳层	S
虚拟相互作用位点	V(或D)

5.2.1 原子类型

每个力场都定义了一组原子类型, 它们具有特征名称或编号, 质量(以a.m.u为单位). 这些原子类型的列表可以在 `atomtypes.atp` (.atp = atom type parameter, 原子类型参数)文件中找到. 因此, 你可以从这个文件开始修改和/或增加原子类型. 下面的示例原子类型来自于 `gromos43a1.ff` 力场

```
O 15.99940 ; carbonyl oxygen (C=O)
OM 15.99940 ; carboxyl oxygen (CO-)
OA 15.99940 ; hydroxyl, sugar or ester oxygen
OW 15.99940 ; water oxygen
N 14.00670 ; peptide nitrogen (N or NH)
NT 14.00670 ; terminal nitrogen (NH2)
NL 14.00670 ; terminal nitrogen (NH3)
NR 14.00670 ; aromatic nitrogen
NZ 14.00670 ; Arg NH (NH2)
NE 14.00670 ; Arg NE (NH)
C 12.01100 ; bare carbon
CH1 13.01900 ; aliphatic or sugar CH-group
```

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

CH2 14.02700 ; aliphatic or sugar CH2-group

CH3 15.03500 ; aliphatic CH3-group

注意: GROMACS以名称来使用原子类型, 而不是编号(参考GROMOS的例子)

5.2.2 虚拟位点

一些力场会使用虚拟相互作用位点(由其他粒子的位置构建的相互作用位点), 并为这些位点设置一些相互作用(例如, 对苯环可产生正确的四极矩), 这些将在4.7中详细介绍.

为了在你的体系中生成虚位点, 你需要在拓扑文件中包含 `[ virtual_sites? ]` 段(为向后兼容也可以使用老版本中的名称 `[ dummies? ]`), 其中 `?` 代表构成虚拟位点的粒子数目, 对类型2为 `2`, 对类型3, 3fd, 3fad和3out为 `3`, 对类型4fdn为 `4`. 类型4fdn取代了旧的4fd类型('type'值为1), 因为它有时会不稳定. 尽管程序内部仍然支持4fd类型, 不过最好不要在新的输入文件中使用. 不同类型将在4.7节解释.

类型2的参数看起来应该像这样:

```
[ virtual_sites2 ]
; Site from funct a
5      1 2      1      0.7439756
```

类型3的像这样:

```
[ virtual_sites3 ]
; Site from funct a b
5      1 2 3      1      0.7439756 0.128012
```

类型3fd的像这样:

```
[ virtual_sites3 ]
; Site from funct a d
5      1 2 3      2      0.5      -0.105
```

类型3afd的像这样:

```
[ virtual_sites3 ]
; Site from funct theta d
5      1 2 3      3      120      0.5
```

类型3out的像这样:

```
; Site from funct a b c
5      1 2 3      4      -0.4      -0.4      6.9281
```

类型4fdn的像这样:

```
[ virtual_sites4 ]
; Site from funct a b c
5      1 2 3 4      2      1.0      0.9      0.105
```

这就构成了虚拟位点, 编号5(第一列 `Site`)基于其他原子的位置, 这些原子的索引号为1和2, 或者1, 2和3, 或者1, 2, 3和4(接下来的第2, 3或4列 `from`), 构建时所遵循的规则由函数类型编号(下一列 `func`)以及具体的参数(最后1, 2或3列 `a b..`)决定. 显然, 原子数目(包括虚拟位点数)取决于分子. 研究一下GROMCAS自带的TIP4P或TIP5P水模型的拓扑文件对理解这些可能很有帮助.

注意: 如果在虚拟位点和/或普通原子之间定义了任何固定的键合相互作用, `grompp` 命令将会移除它们(除非使用 `-normvsbds` 选项). 键合相互作用的移除是在生成排除之后, 因为生成排除是以“化学”的键相互作用为基础.

可以使用更普遍的方法构建虚拟位点, 利用的指令是 `[ virtual_sustesn ]`. 所需要的参数列在表5.5中. 下面是将一组给定原子的几何中心定义为虚拟位点的例子:

```
[ virtual_sitesn ]
; Site funct from
5      1      1 2 3 4
```

表 5.2: GROMACS中的静态原子类型性质

性质	符号	单位
----	----	----

- 5.1 简介
- 5.2 粒子类型
 - 5.2.1 原子类型
 - 5.2.2 虚拟位点
- 5.3 参数文件
 - 5.3.1 原子
 - 5.3.2 非键参数
 - 5.3.3 键参数
 - 5.3.4 分子内的对相互作用
 - 5.3.5 隐式溶剂化模型
- 5.4 排除
- 5.5 约束算法
- 输入文件
 - 5.6.1 残基数据库
 - 5.6.2 残基构建单元数据库
 - 5.6.3 原子重命名数据库
 - 5.6.4 氢数据库
 - 5.6.5 末端数据库
 - 5.6.6 虚拟位点数据库
 - 5.6.7 特殊键
- 5.7 文件格式
 - 5.7.1 拓扑文件
 - 5.7.2 Molecule.itp文件
 - 5.7.3 ifdef语句
 - 5.7.4 用于自由能计算的拓扑文件
 - 5.7.5 约束力
 - 5.7.6 坐标文件
- 5.8 力场的组织
 - 5.8.1 力场文件
 - 5.8.2 改变力场参数
 - 5.8.3 添加原子类型

类型	-	-
质量	m	a.m.u
电荷	q	e
epsilon	ε	kJ/mol
sigma	σ	nm

5.3 参数文件

5.3.1 原子

原子类型 **静态性质** (参见表 5.2)的指定基于几个地方的数据. 质量来源于 `atomtypes.apf` 文件(参见5.2.1节), 电荷来源于 `*.rtp` (.rtp = residue topology parameter, 残基拓扑参数, 参见5.6.1节)文件. 这就意味着只对构建氨基酸, 核酸的基本单元定义了电荷, 对其他的分子, 用户需要自己定义电荷. 当使用 `pdb2gmx` 程序生成一个拓扑文件(`*.top`)时, 来自这些文件的信息将被整合在一起.

5.3.2 非键参数

非键参数包括van der Waals参数V(`c6` 或 σ , 由组合规则决定)和W(`c12` 或 ϵ), 它们列在 `ffnonbonded.itp` 文件中, 其中的 `ptype` 是粒子类型(参见表 5.1). `[ *type ]` 指令中的条目和键合参数会被应用它们在拓扑文件中的相应部分. 除了将在5.3.4节中提到的那些, 缺少参数将导致警告.

```
[ atomtypes ]
; name      at.num      mass      charge      ptype      V(c6)      W(
      O          8      15.99940    0.000        A      0.22617E-02    0.74158
      OM         8      15.99940    0.000        A      0.22617E-02    0.74158
      .....

[ nonbond_params ]
; i      j      func      V(c6)      W(c12)
      0      0      1      0.22617E-02    0.74158E-06
      0      OA     1      0.22617E-02    0.13807E-05
      .....
<----->
```

注意: GROMACS所包含的大部分力场也带有 `at.num.` 列, 但相同的信息位于OPLS_AA `bond_type` 列. 参数V与W的含义取决于拓扑文件 `[ defaults ]` 段中选择的组合规则(参见5.7.1节).

对组合规则1:

$$\begin{aligned} V_{ii} &= C_i^{(6)} = 4\epsilon_i \sigma_i^6 [\text{kJ mol}^{-1} \text{nm}^6] \\ W_{ii} &= C_i^{(12)} = 4\epsilon_i \sigma_i^{12} [\text{kJ mol}^{-1} \text{nm}^{12}] \end{aligned} \quad (5.1)$$

对组合规则2和3:

$$\begin{aligned} V_{ii} &= \sigma_i [\text{nm}] \\ W_{ii} &= \epsilon_i [\text{kJ mol}^{-1}] \end{aligned} \quad (5.2)$$

对不同原子类型间的一些或所有组合都可以在 `[ nonbond_params ]` 段给出, 参数V与W的定义同上. 对于其他没有给出的任何原子组合, 将根据组合原则, 利用相应原子类型的参数进行计算.

对组合规则1和3:

$$\begin{aligned} C_{ij}^{(6)} &= \sqrt{C_i^{(6)} C_j^{(6)}} \\ C_{ij}^{(12)} &= \sqrt{C_i^{(12)} C_j^{(12)}} \end{aligned} \quad (5.3)$$

对组合规则2:

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

$$\sigma_{ij} = \frac{1}{2}(\sigma_i + \sigma_j) \quad (5.4)$$

$$\epsilon_{ij} = \sqrt{\epsilon_i \epsilon_j}$$

当需要提供 σ 和 ϵ 时(规则2和3), 看起来不可能使用非零的 C^{12} 与零值的 C^6 参数进行组合. 然而, 提供负值的 σ 恰好会这样做, C^6 被设为零, C^{12} 正常计算. 这只是代表了读入 σ 值的一种特殊情况, 没有其他的.

对Buckingham势只有一种组合规则:

$$A_{ij} = \sqrt{A_{ii} A_{jj}}$$

$$B_{ij} = 2 / \left(\frac{1}{B_{ii}} + \frac{1}{B_{jj}} \right) \quad (5.5)$$

$$C_{ij} = \sqrt{C_{ii} C_{jj}}$$

5.5.3 键参数

键参数(如, 键长, 键角, 异常/正常二面角)都列在 `ffbonded.itp` 文件中. 这个数据库中的条目分别给出了相互作用的原子类型, 相互作用的类型, 与相互作用有关的参数. 在处理拓扑文件时, `grompp` 程序会读取这些参数, 然后应用到相关的键参数中. 例如将 `bondtypes` 应用到 `[ bonds ]` 指令中的条目, 等等. 从相关的 `[ *type ]` 指令中缺失的任何键参数都会导致致命错误. 相互作用的类型列于表 5.5. 下面是从这些文件中摘录的例子:

```
[ bondtypes ]
; i   j   func      b0      kb
  C   O       1    0.12300   502080.
  C  OM       1    0.12500   418400.
.....

[ angletypes ]
; i   j   k   func      th0      cth
HO   OA   C       1    109.500   397.480
HO   OA  CH1      1    109.500   397.480
.....

[ dihedraltypes ]
; i   l   func      q0      cq
NR5*  NR5       2    0.000   167.360
NR5*  NR5*      2    0.000   167.360
.....

[ dihedraltypes ]
; j   k   func      phi0      cp      mult
  C   OA       1    180.000   16.736      2
  C   N       1    180.000   33.472      2
.....
```

```
[ dihedraltypes ]

;
; Ryckaert-Bellemans Dihedrals
;
; aj   ak  funct
CP2  CP2      3    9.2789  12.156  -13.120  -3.0597  26.240  -31.495
```

在 `ffbonded.itp` 文件中, 你可以添加键参数. 如果你想为新的原子类型增加参数, 请确保你已经在 `atomtypes.atp` 中定义了它们.

5.3.4 分子内的对相互作用

分子中原子对之间额外的Lennard-Jones和静电相互作用可以添加到分子定义部分的 `[ pairs ]` 段中. 这些相互作用的参数可以独立于非键相互作用参数进行设置. 在GROMOS力场中, `[ pairs ]` 仅仅用于修改1-4相互作用(相隔3条键的两个原子之间的相互作用). 在这些力场中, 1-4相互作用并不包括在非键相互作用中(参见5.4节).

- 5.1 简介
- 5.2 粒子类型
 - 5.2.1 原子类型
 - 5.2.2 虚拟位点
- 5.3 参数文件
 - 5.3.1 原子
 - 5.3.2 非键参数
 - 5.3.3 键参数
 - 5.3.4 分子内的对相互作用
 - 5.3.5 隐式溶剂化模型
- 5.4 排除
- 5.5 约束算法
- 输入文件
 - 5.6.1 残基数据库
 - 5.6.2 残基构建单元数据库
 - 5.6.3 原子重命名数据库
 - 5.6.4 氢数据库
 - 5.6.5 末端数据库
 - 5.6.6 虚拟位点数据库
 - 5.6.7 特殊键
- 5.7 文件格式
 - 5.7.1 拓扑文件
 - 5.7.2 Molecule.itp文件
 - 5.7.3 ifdef语句
 - 5.7.4 用于自由能计算的拓扑文件
 - 5.7.5 约束力
 - 5.7.6 坐标文件
- 5.8 力场的组织
 - 5.8.1 力场文件
 - 5.8.2 改变力场参数
 - 5.8.3 添加原子类型

```
[ pairtypes ]
; i      j      func          cs6          cs12 ; THESE ARE 1-4 INTERACTION
  0      0      1      0.22617E-02      0.74158E-06
  0      OM     1      0.22617E-02      0.74158E-06
  ....
```

`ffnonbonded.itp` 文件中原子类型的对相互作用参数位于 `[ pairtypes ]` 段。GROMOS力场显式地列出了所有这些相互作用的参数, 但对于OPLS这样的力场这一段可能是空的, 因为这些力场通过统一地缩放参数来计算1-4相互作用。对于那些不在 `[ pairtypes ]` 段出现的对参数, 只能当 `forcefield.itp` 文件中 `[ defaults ]` 指令的 `gen-pair` 设置为“yes”时才能生成(参见5.7.1节)。当 `gen-pairs` 设置为“no”时, `grompp` 程序会对每个未设定参数的对类型产生警告。

普通对相互作用和1-4相互作用的函数类型为1。函数类型2和 `[ pairs_nb ]` 特别用于自由能模拟。当计算水合自由能时, 需要将溶质与溶剂去耦合。这可通过添加一个B-状态拓扑(参见3.12节)实现, 其中所有溶质的非键参数, 即电荷和LJ参数, 都被设置为0。然而, A状态和B状态之间自由能的差值并不是总的水合自由能。我们必须通过重新引入真空中溶质分子内部的库伦和LJ相互作用来增加自由能。当溶质内的库伦和LJ相互作用未修改时, 第二步可以与第一步结合起来。为此, 引入了对函数类型2, 它与函数类型1完全相同, 除了B状态参数与A状态参数始终相同。

在 `[ pairtypes ]` 段搜索参数, 函数类型1和2之间并没有什么差别。对相互作用段 `[ pair_nb ]` 用于取代非键相互作用。它使用未缩放的电荷与非键LJ参数, 并只使用A状态的参数。**注意** 要为 `[ pairs_nb ]` 中列出的所有原子对添加排除, 否则这些原子对也将正在正常的邻居列表中结束。

作为替代, 通过使用 `couple-moltype`, `couple-lambda0`, `couple-lambda1`, 和 `couple-intramol` 关键字, 我们可以不修改拓扑文件而获得同样的行为。具体请参考3.12节和6.1节。

所有这三种对类型全都使用普通的库伦作用, 即便当使用反应场, PME, Ewald或移位库伦相互作用来计算非键相互作用时。类型1和2的能量会写入能量和日志文件, 其中每个能量组对都有单独的“LJ-14”和“Coulomb-14”项。 `[ pairs_nb ]` 的能量会添加到“LJ-(SR)”和“Coulomb-(SR)”项中。

5.3.5 隐式溶剂化模型

GROMACS自4.5版本起开始支持隐式溶剂。拓扑文件中引入了一个节段用于列出相关参数:

```
[ implicit_genborn_params ]
; Atomtype      sar      st      pi      gbr      hct
  NH1           0.155    1      1.028    0.17063    0.79 ; N
  N             0.155    1      1      0.155     0.79 ; Proline backbone N
  H             0.1      1      1      0.115     0.85 ; H
  CT1           0.180    1      1.276    0.190     0.72 ; C
```

在上面的例子中, 先列出了原子类型, 后面有5个数字和注释(分号后面)。

目前并未使用1-3列的值。它们与更精细的表面积算法, 特别是Qiu等提出的一个算法有关[70]。第4列为原子的范德华半径, 用于计算Born半径。电介质偏移值在 `*.mdp` 文件中指定, 对不同的计算Born半径的方法, 程序会将输入的范德华半径减去此偏移值, 见Onufriev等的说明[72]。第5列是HCT和OBC模型的缩放因子, 数值来源于Tinker程序实现的HCT成对缩放方法[71]。这种方法经过修改, 缩放因子调整后使得解析的表面积和使用HCT算法计算的GB表面积之间的偏差最小。根据Hawkins[71]等的建议, 缩放不适用于成对的, 而只能基于每个原子, 因此对它们进行了进一步的修正。

5.4 排除

`grompp` 程序会对相邻直到一定数目键的原子生成非键相互作用的排除, 如在拓扑文件的 `[ moleculetype ]` 段中定义的那样(参见5.7.1节)。当彼此之间以“化学”键(`[ bonds ]`

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

类型1到5, 7或8)或约束([constraints] 类型1)连接时, 粒子被认为是键合在一起的. 类型5 [bonds] 可用于创建两个原子之间无相互作用的连接. 有一种不通过化学键连接原子的简谐相互作用([bonds] 类型6), 也有一种不通过化学键连接原子而固定距离的第二类约束类型([constraints] 类型2). 所有这些相互作用的完整列表见表 5.5.

分子内额外的排除可以手动在 [exclusions] 段中添加. 每行必须以一个原子编号开始, 后面跟着一个或多个原子编号. 第一个原子和其他原子之间的所有非键相互作用都会被排除.

当需要排除原子组内部或彼此之间的非键相互作用时, 使用能量监测组排除会更方便和高效(参见3.3节).

5.5 约束算法

约束的定义在 [constraints] 段中, 其格式为两个原子编号, 后面跟着函数类型和约束距离. 函数类型可以为1或2, 它们之间的唯一区别在于, 类型1用于产生排除而类型2不产生排除(参见5.4节). 距离是通过在 *.mdp 文件中选择的LINCS或SHAKE算法进行约束的. 在自由能计算中, 通过增加第二个约束距离, 这两种类型的约束都可以进行微扰(参见5.7.5节). grompp 程序可自动将一些类型的键或键角(参见表 5.5)转变为约束, 在 *.mdp 文件中有很多相关的选项.

我们也实现了SETTLE算法[45], 它对SHAKE进行解析求解, 只用于水. 可以在拓扑文件中选择SETTLE. 例如, 请参看SPC水分子的定义:

```
[ moleculetype ]
; molname      nrexcl
  SOL           1

[ atoms ]
; nr      at type      res nr      ren nm      at nm      cg nr      charge
  1        OW           1        SOL        OW1        1        -0.82
  2        HW           1        SOL        HW2        1        0.41
  3        HW           1        SOL        HW3        1        0.41

[ settles ]
; OW      funct      doh      dhh
  1        1          0.1      0.16333

[ exclusions ]
1      2      3
2      1      3
3      1      2
```

[settles] 指令定义了水分子的第一个原子, SETTLE函数类型始终为1, 必须给出O-H和H-H之间的距离. 注意, SETTLE算法也可用于TIP3P和TIP4P水分子模型[125]. TIP3P具有另一个构型, TIP4P具有一个虚拟位点, 但由于是生成的, 不需要抖动(或扰动).

输入文件

GROMACS的 pdb2gmx 程序可以根据输入的坐标文件产生拓扑文件, 它支持几种不同格式的坐标文件, 但 *.pdb 是最常用的(也因此程序名称为 pdb2gmx). pdb2gmx 程序运行时会在GROMACS share/top 目录的子目录和你的工作目录中搜索力场, 并根据扩展名为 .ff 的目录中的 forcefield.itp 文件识别力场. 目录中可能存在 forcefield.doc 文件, 如果存在, pdb2gmx 会将此文件的第一行作为力场的简单描述显示给用户, 以帮助用户选择力场. 否则, 用户可以使用 pdb2gmx 的命令行参数 -ff xxx 来选择力场, 所选的力场将位于 xxx.ff 目录. 搜索力场时, pdb2gmx 会首先搜索工作目录, 然后再搜索GROMACS的 share/top 目录, 并使用找到的第一个匹配 xxx.ff 的目录.

pdb2gmx 会读入两个通用文件: 位于力场目录的原子类型文件(扩展名为 .atp, 参见5.2.1节), 位于工作目录或GROMACS share/top 目录的 residuetypes.dat 文件. residuetypes.dat 文件决定了哪些残基名称将被视为蛋白质, DNA, RNA, 水和离子.

对不同类型的分子, pdb2gmx 可以读入一个或多个数据库及拓扑信息. 属于同一个数据库的一组文件应具有相同的基准名称(basename), 基准名称最好能够对分子类型有所说明

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

(如, 氨基酸, rna, dna). 可能的文件如下:

- `<basename>.rtp`
- `<basename>.r2b` (可选)
- `<basename>.arn` (可选)
- `<basename>.hdb` (可选)
- `<basename>.n.tdb` (可选)
- `<basename>.c.tdb` (可选)

只有包含了构建单元拓扑信息的 `.rtp` 文件是必需的, 来自于其他文件的信息只用于具有相同基准名的 `.rtp` 文件中的构建单元. 通过在工作目录中放置具有相同基准名的额外文件, 用户也可以为力场添加新的构建单元. 默认只能定义一个额外的构建单元, 但使用 `-rtpo` 选项调用 `pdb2gm` 程序时, 可以使用本地文件中的构建单元来代替力场中默认的构建单元.

5.6.1 残基数据库

残基数据库文件的扩展名为 `.rtp`. 这个文件原本包含蛋白质的构建单元(氨基酸), 是 GROMACS对GROMOS `rt37c4.dat` 文件的解释说明, 因此残基数据库文件包含常用构建单元的信息(键, 电荷, 电荷组, 异常二面角). 最好 **不要** 更改这个文件, 因为它是 `pdb2gm` 的标准输入. 但如果确实需要修改, 请修改工作目录中的 `*.top` 文件(参见5.7.1节), 或 `.rtp` 文件, 像5.6节解释的那样. 通过直接包含 `*.itp` 拓扑文件定义一个新的小分子的拓扑可能更容易一些, 具体作法将在5.7.2节讨论. 当添加新的蛋白质残基到数据库中时, 别忘了将残基名称添加到 `residuetypes.dat` 文件中, 这样 `grompp`, `make_ndx` 和分析程序才能将残基识别为蛋白质残基(参见8.1.1节).

`.rtp` 文件只会被 `pdb2gm` 程序使用. 如前面提到的, `pdb2gm` 程序仅仅从 `.rtp` 数据库中读入键, 原子电荷, 电荷组和异常二面角这些额外信息, 其余的信息是从坐标输入文件读入的. 一些蛋白质的坐标文件中包含非标准氨基酸. 你必须为这些“陌生”残基创建构建单元, 否则你将无法得到 `*.top` 文件. 对坐标文件中的一些分子, 如配体, 多原子离子, 晶格化溶剂分子等, 也是一样. 残基数据库可以根据以下方法创建:

```
[ bondedtypes ] ; mandatory 必需
; bonds      angles      dihedrals      impropers
      1          1          1              2 ; mandatory 必需
```

```
[ GLY ] ; mandatory 必需
```

```
[ atoms ] ; mandatory 必需
; name      type      charge      chargegroup
      N        N       -0.280        0
      H        H        0.280        0
      CA       CH2       0.000        1
      C        C        0.380        2
      O        O       -0.380        2
```

```
[ bonds ] ; optional 可选
; atom1      atom2      b0      kb
      N        H
      N        CA
      CA       C
      C        O
      -C       N
```

```
[ exclusions ] ; optional 可选
; atom1      atom2
```

```
[ angles ] ; optional 可选
; atom1      atom2      atom3      th0      cth
```

```
[ dihedrals ] ; optional 可选
; atom1      atom2      atom3      atom4      phi0      cp      mult
```

```
[ impropers ] : optional 可选
```

- 5.1 简介
- 5.2 粒子类型
 - 5.2.1 原子类型
 - 5.2.2 虚拟位点
- 5.3 参数文件
 - 5.3.1 原子
 - 5.3.2 非键参数
 - 5.3.3 键参数
 - 5.3.4 分子内的对相互作用
 - 5.3.5 隐式溶剂化模型
- 5.4 排除
- 5.5 约束算法
- 输入文件
 - 5.6.1 残基数据库
 - 5.6.2 残基构建单元数据库
 - 5.6.3 原子重命名数据库
 - 5.6.4 氢数据库
 - 5.6.5 末端数据库
 - 5.6.6 虚拟位点数据库
 - 5.6.7 特殊键
- 5.7 文件格式
 - 5.7.1 拓扑文件
 - 5.7.2 Molecule.itp文件
 - 5.7.3 ifdef语句
 - 5.7.4 用于自由能计算的拓扑文件
 - 5.7.5 约束力
 - 5.7.6 坐标文件
- 5.8 力场的组织
 - 5.8.1 力场文件
 - 5.8.2 改变力场参数
 - 5.8.3 添加原子类型

```

;atom1      atom2      atom3      atom4      q0      cq
          N          -C          CA          H
          -C          -CA          N          -O

[ ZN ]

[ atoms ]
      ZN      ZN      2.000      0
    
```

文件是自由格式, 唯一的限制是每行最多一个条目. 文件中的第一个域为 `[ bondedtypes ]`, 后面跟着四个数字, 分别代表键, 键角, 二面角和异常二面角的相互作用类型. 文件中的残基条目包含原子和(可选的)键, 键角, 二面角, 异常二面角. 电荷组代码代表电荷组的编号, 相同电荷组的原子应该始终按顺序排列. 当使用 `pdb2gmx` 程序及氢数据库添加缺失的氢原子时(参见5.6.4), `.rtp` 条目中定义的原子名称应该精确地对应于氢数据库中使用的名称约定. 键相互作用中的原子名称前可添加减号或加号, 分别代表原子处于残基之前或残基之后. 添加到键, 键角, 二面角和异常二面角的显式参数会覆盖 `.itp` 文件中的标准参数, 但只能用于特殊情况. 除参数外, 也可可为每个键相互作用添加字符串. GROMOS-96的 `.rtp` 文件就是这样. 这些字符串会被复制到拓扑文件, 通过使用 `grompp` C预处理器的 `#define` 语句, 可用力场参数替换这些字符串.

`pdb2gmx` 程序会自动生成所有的键角, 这意味着对大多数力场 `[ angles ]` 域仅用于覆盖 `.itp` 参数. 对GROMOS-96力场必须指定所有键角的相互作用编号.

`pdb2gmx` 程序会自动为每个旋转键生成一个正常二面角, 倾向位于重原子上. 当使用 `[ dihedrals ]` 域时, 不会为对应于指定二面角的键生成其他二面角. 可以为一条旋转键指定多个二面角函数. 对CHARMM27力场, 使用 `pdb2gmx` 程序默认的 `-cmap` 选项可为二面角增加校正图. 详细信息请参考4.10.4节.

`pdb2gmx` 会设置排除数为3, 这意味着最多以3条键连接的原子之间的相互作用都被排除了. 程序会为相隔三条键的所有原子对生成对相互作用(氢原子除外). 当需要排除更多的相互作用, 或不需要生成一些对相互作用时, 可添加 `[ exclusions ]` 域, 后面跟着位于不同行上的原子名称对, 这些原子之间的所有非键和对相互作用都将被排除.

5.6.2 残基构建单元数据库

每个力场对残基都有自己的命名约定. 大部分残基都具有一致的命名, 但有一些残基, 特别是那些具有不同质子化状态的残基, 可能具有许多不同的名称. `.r2b` 文件可用于将标准的残基名称转换为力场构建单元名称.

表 5.3 GROMACS内部的残基名称约定

简写	英文名称	含义
ARG	protonated arginine	质子化精氨酸
ARGN	neutral arginine	中性精氨酸
ASP	negatively charged aspartic acid	带负电的天冬氨酸
ASPH	neutral aspartic acid	中性天冬氨酸
CYS	neutral cysteine	中性半胱氨酸
CYS2	cysteine with sulfur bound to another cysteine or a heme	通过硫与另一个半胱氨酸或血红素结合的半胱氨酸
GLU	negatively charged glutamic acid	带负电的谷氨酸
GLUH	neutral glutamic acid	中性谷氨酸
HISD	neutral histidine with N_{δ} protonated	N_{δ} 质子化的中性组氨酸
HISE	neutral histidine with N_{ϵ} protonated	N_{ϵ} 质子化的中性组氨酸
HISH	positive histidine with both N_{δ} and N_{ϵ} protonated	N_{δ} 和 N_{ϵ} 质子化的带正电的组氨酸
HIS1	histidine bound to a heme	结合到血红素的组氨酸
LYSN	neutral lysine	中性赖氨酸
LYS	protonated lysine	质子化赖氨酸

目录
5.1 简介
5.2 粒子类型
5.2.1 原子类型
5.2.2 虚拟位点
5.3 参数文件
5.3.1 原子
5.3.2 非键参数
5.3.3 键参数
5.3.4 分子内的对相互作用
5.3.5 隐式溶剂化模型
5.4 排除
5.5 约束算法
输入文件
5.6.1 残基数据库
5.6.2 残基构建单元数据库
5.6.3 原子重命名数据库
5.6.4 氢数据库
5.6.5 末端数据库
5.6.6 虚拟位点数据库
5.6.7 特殊键
5.7 文件格式
5.7.1 拓扑文件
5.7.2 Molecule.itp文件
5.7.3 ifdef语句
5.7.4 用于自由能计算的拓扑文件
5.7.5 约束力
5.7.6 坐标文件
5.8 力场的组织
5.8.1 力场文件
5.8.2 改变力场参数
5.8.3 添加原子类型

HEME	heme	血红素
------	------	-----

如果力场目录中不存在 `.r2b` 文件, 或残基未被列出, 会假定构建单元的名称与残基名称相同. `.r2b` 文件可包含2或5列. 2列格式为, 第一列为残基名称, 第二列为构建单元名称. 5列格式具有3个附加列, 分别为出现在N端, C端和同时出现在两个末端的残基(单个残基分子)的构建单元. 这对一些力场有用, 例如AMBER力场. 如果不存在一个或多个末端, 应在相应的列中输入短划线.

对残基, 存在GROMACS的命名约定, 此约定仅通过 `.r2b` 文件和 `specbond.dat` 文件显现出来(除 `pdb2gmx` 代码外). 只有当你添加残基类型到 `.rtp` 文件时, 这个约定是才变得重要. 此约定列在表5.3中. 对于特殊的键, 如与血红素基团相连的键, GROMACS命名约定通过 `specbond.dat` 引入(参见5.6.7节), 如果需要, 此约定随后可以利用 `.r2b` 文件进行翻译.

5.6.3 原子重命名数据库

力场中使用的原子名称经常不遵循IUPAC或PDB约定. `.arn` 数据库用于将坐标文件中的原子名称转换为力场中的名称. 未列出的原子会保持原有名称. `.arn` 文件有三列, 分别为构建单元名称, 旧的原子名称和新的原子名称. 残基名称支持问号通配符, 用以匹配单个字符.

`share/top` 目录下还存在一个通用的原子重命名文件 `xlateat.dat`, 可以将坐标文件中常见的非标准原子名称转换为IUPAC/PDB约定名称. 因此, 当编写力场文件时, 你可以使用标准的原子名称, 除了将其翻译为力场名称外, 不需要进一步的翻译.

5.6.4 氢数据库

氢数据库储存在 `.hdb` 文件中, 它包含了关于 `pdb2gmx` 程序如何将氢原子连接到已有原子的信息. 在GROMACS 3.3版本以前的数据库中, 根据连接的原子对氢原子进行命名: 把连接原子名称的首字母用'H'代替. 从3.3版本开始, 必须明确列出氢原子, 因为以前的做法仅适用于蛋白质, 因而不能推广用于其它分子. 如果有一个以上的氢原子连接到相同的原子, 氢原子名称的末尾将添加一个数字. 例如, 添加两个氢原子到 `ND2` (天冬酰胺), 氢原子将被命名为 `HD21` 和 `HD22`. 这很重要, 因为 `.rtp` 文件(参见5.6.1节)中的命名必须相同. 氢数据库的格式如下:

; res	# additions					
	# H	add	type	H	i	j k
ALA	1					
	1	1		H	N	-C CA
ARG	4					
	1	2		H	N	CA C
	1	1		HE	NE	CD CZ
	2	3		HH1	NH1	CZ NE
	2	3		HH2	NH2	CZ NE

第一行为残基名称(ALA或ARG)以及氢原子的类型数, 这些氢原子可根据氢数据库添加到残基中. 后面的行每行对应于一个氢原子的添加:

- 添加的氢原子数
- 添加氢原子的方法, 可能的方法如下:

1 单个平面氢原子, 如环或肽键

产生一个氢原子(n), 置于原子(i,j,k)形成的平面内, 位于角(j-i-k)的平分线上, 距离原子i 0.1 nm, 并使角(n-i-j)和(n-i-k)大于90度.

2 单个氢原子, 例如羟基

产生一个氢原子(n), 距离原子i 0.1 nm, 并使角(n-i-j)为109.5度, 二面角(n-i-j-k)为反式.

3 两个平面氢原子, 如乙烯 $-C=CH_2$ 或酰胺 $-C(=O)NH_2$

产生两个氢原子(n1,n2), 距离原子i 0.1 nm, 并使角(n1-i-j)与(n2-i-j)为120度, 二面角(n1-i-j-k)为顺式, (n2-i-j-k)为反式, 这样命名符合IUPAC标准[126].

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

4 两个或三个四面体氢原子, 例如-CH₃

产生三个(n1,n2,n3)或两个(n1,n2)氢原子, 距离原子i 0.1 nm, 并使角(n1-i-j), (n2-i-j), (n3-i-j)都为109.47度, 二面角(n1-i-j-k)为反式, (n2-i-j-k)为反式+120度, (n3-i-j-k)=反式+240度.

5 单个四面体氢原子, 例如C₃CH

产生一个氢原子(n'), 距离原子i 0.1 nm, 处于四面体构型, 角(n'-i-j), (n'-i-k)和(n'-i-l)都为109.47度.

6 两个四面体氢原子, 如C-CH₂-C

产生两个氢原子(n1,n2), 距离原子i 0.1 nm, 处于四面体构型, 位于平分角j-i-k的平面上, 并且角(n1-i-n2), (n1-i-j)和(n1-i-k)都为109.47度.

7 两个水中的氢原子

根据SPC[81]水模型的几何构型在原子i周围产生两个氢原子. 对称轴交替位于三个坐标轴两个方向之间.

10 三个水中的“氢原子”

根据SPC[81]水模型的几何构型在原子i周围产生两个氢原子. 对称轴交替位于三个坐标轴两个方向之间. 此外, 在氧原子位置产生一个额外的粒子, 并将其名称的第一个字母以'M'替代. 此方法用于四位点的水模型, 如TIP4P[125].

11 四个水中的“氢原子”

同上, 但会在氧原子位置产生两个额外的粒子, 名字分别为'LP1'和'LP2'. 此方法用于五位点的水模型, 如TIP5P[127].

- 新的氢原子的名称(或其前缀, 如 **HD2**, 对前面天冬酰胺的例子).
- 三或四个控制原子(i,j,k,l), 其中的第一个始终为与氢原子相连的原子, 另外两个或三个取决于所选的代码. 对于水, 只有一个控制原子.

对一些非常特殊的情况, 可以利用上面的方法近似地解决, 并进行适当的能量最小化, 这样得到的构型用于MD模拟的初始构型已经足够好了. 例如对仲胺氢, 亚硝酰基氢(C=NH)甚至乙炔氢都可利用上面方法2羟基氢进行近似的添加.

5.6.5 末端数据库

末端数据库储存在 **aminoacids.n.tdb** 和 **aminoacids.c.tdb** 文件中, 分别对应于N端和C端. 文件包含了关于 **pdb2gmx** 程序如何将原子连接到已有原子, 应该删除或更改哪些原子, 应该添加哪些键相互作用的信息. 文件的格式如下(来源于

gromos43a1.ff/aminoacids.c.tdb):

```
[ None ]

[ COO- ]
[ replace ]
C      C      C      12.011      0.27
O      O1     OM      15.9994     -0.635
OXT     O2     OM      15.9994     -0.635
[ add ]
2      8      O      C      CA      N
OM      15.9994     -0.635
[ bonds ]
C O1 gb_5
C O2 gb_5
[ angles ]
O1 C O2 ga_37
CA C O1 ga_21
CA C O2 ga_21
[ dihedrals ]
N CA C O2 gd_20
[ impropers ]
C CA O2 O1 gi_1
```

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

该文件以块为组织, 每块的标题指定了块的名称. 这些块对应于可添加到分子的不同类型的末端. 在本例中 [COO-] 为第一块, 对应于将末端碳原子更改为去质子化的羧基. [None] 为第二末端类型, 对应于维持分子原样的末端. 块名称不能取以下的任何一种: `replace`, `add`, `delete`, `bonds`, `angles`, `dihedrals`, `impropers`. 否则会干扰块的参数, 并可能令读者十分迷惑.

每个块可使用以下选项:

- [`replace`]

将一个已有原子替换为具有不同原子类型, 原子名称, 电荷和/或质量的原子. 此项可用于替换输入坐标文件和 `.rtp` 数据库中的原子, 但也可用于仅仅重命名输入坐标中的原子, 以使它匹配力场中的名称. 对后一种情况, 也应该存在相应的 [`add`] 段, 用于指示如何添加相同的原子, 这样才能知道序列中的位置和成键. 这种原子可出现在输入坐标中并保持, 或不出现在输入坐标中而是通过 `pdb2gmx` 程序构建. 对于每一个要替换的原子, 应该输入具有以下域的一行:

- 要替换原子的名称
- 新的原子名称(可选)
- 新的原子类型
- 新的质量
- 新的电荷

- [`add`]

添加新的原子. 对于每一个(组)要添加的原子需要两行输入. 第一行包含与氢数据库相同的域(新原子的名称, 原子数目, 添加类型, 控制原子, 参见5.6.4节), 但增加了两个只用于C端的添加类型

8 双羧基氧原子, -COO⁻

根据规则3产生两个氧原子(n1,n2), 距离原子i 0.136 nm, 角(n1-i-j)和(n2-i-j)为117度

9 羧基氧原子和氢原子, -COOH

根据规则3产生两个氧原子(n1,n2), n1和n2分别距离原子i 0.123 nm和0.125 nm, 角(n1-i-j)为121度, (n2-i-j)为115度. 根据规则2在n2周围产生一个氢原子(n'), 其中n'-n2-i和n'-n2-i-j应分别对应于n-i-j和n-i-j-k.

此行之后, 下面的一行指定了添加原子的细节, 与替换原子的方式相同, 即:

- 原子类型
- 质量
- 电荷
- 电荷组(可选)

如氢数据库(参见5.6.1节)一样, 当一个以上的原子连接到一个已有原子时, 原子名称的末尾会追加一个数字. **注意**, 像在氢数据库中一样, 原子名称现在与控制原子处于同一行, 而在GROMACS 3.3版本之前它位于第二行的开始. 当电荷组域空白时, 添加的原子和它连接的原子将具有相同的电荷组编号.

- [`delete`]

删除已有的原子. 每个原子名称一行.

- [`bonds`], [`angles`], [`dihedrals`] 和 [`impropers`]

添加额外的键参数. 格式与 `*.rtp` 文件中使用的相同, 参见5.6.1节.

5.6.6 虚拟位点数据库

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

由于不能依赖输入文件中氢的位置, 我们需要一个特殊的输入文件以确定要添加的虚拟氢位点的几何构型和参数. 为创建更复杂的虚拟位点(例如当保持整个芳香侧链刚性时), 我们还需要关于平衡键长和侧链中所有原子角度的信息. 对每个力场, 这些信息在 `.vsd` 文件中指定. 与末端类似, 对 `.rtp` 文件中每个残基类型都有一个这样的文件.

虚拟位点数据库真的不是一个非常简单的信息列表. 它开始的两段指定了用于 CH_3 , NH_3 和 NH_2 基团的质量中心(通常称为 $\text{MCH}_3/\text{MNH}_3$). 根据氢原子和重原子之间的平衡键长和键角, 我们需要在这些质量中心之间施加略有不同的约束距离. **注意**, 我们 **不** 需要指定实际的参数(会自动生成), 而只需要指定使用的质量中心类型. 为此, 有三个段名称 `[ CH3 ]`, `[ NH3 ]` 和 `[ NH2 ]`. 对每个段我们需要三列. 第一列为连接到2/3氢原子的原子类型, 第二列为连接的下一个重原子的类型, 第三列为使用的质量中心类型. 作为一种特殊情况, 在 `[ NH2 ]` 段的第二列也可以指定为 `planar`, 代表将使用不同的构造方法, 而不使用质量中心. 对于 NH_2 基团是否应为平面, 目前的一些力场观点不一, 但我们力图保持力场默认的平衡参数.

虚拟位点数据库的第二部分明确地包含芳香族侧链的原子对/三联组的平衡键长和键角. 目前, 虚拟位点生成代码的特定例会读入这些项, 因此如果你想扩展它, 如扩展到核酸, 你需要编写新的代码. 这些段根据氨基酸的短名称进行命名(`[ PHE ]`, `[ TYR ]`, `[ TRP ]`, `[ HID ]`, `[ HIE ]`, `[ HIP ]`), 简单地包含2或3列原子名称, 接着是指定键长(以nm为单位)或键角(以度为单位)的数字. **注意**, 对整个分子的平衡几何构型有所近似, 如果分子未处于自然状态, 其单个键长/键角的值可能与平衡值不同.

5.6.7 特殊键

`pdb2gmx` 程序生成残基间化学键时使用的主要机制为, 从头到尾连接不同残基中的骨架原子进而形成大分子. 在某些情况下(如二硫键, 血红素基团, 支化聚合物), 有必要创建非骨架残基间的化学键. `specbond.dat` 文件即用于此功能. 残基必须属于相同的 `[ moleculetype ]`. 当操纵不同链之间的特殊残基间的化学键时, `pdb2gmx` 程序的 `-merge` 和 `-chainsep` 功能很有用.

`specbond.dat` 文件的第一行表示文件中的条目数. 如果你添加了一个新的条目, 请确保增加此数字. 文件的其余行提供创建键的具体说明. 行的格式如下:

```
resA atomA nbondsA resB atomB nbondsB length newresA newresB
```

分别代表:

1. `resA` 参与成键的残基A的名称.
2. `atomA` 残基A中形成键的原子的名称.
3. `nbondsA` `atomA` 可以形成的键的总数.
4. `resB` 参与成键的残基B的名称.
5. `atomB` 残基B中形成键的原子的名称.
6. `nbondsB` `atomB` 可以形成的键的总数.
7. `length` 键的参考长度. 在提供给 `pdb2gmx` 程序的坐标文件中, 若 `atomA` 和 `atomB` 之间的距离不在 `length` $\pm 10\%$ 范围内, 它们之间不会形成键.
8. `newresA` 残基A的新名称, 如果必要. 有些力场对与二硫键或血红素相连的半胱氨酸使用了CYS2之类的名称.
9. `newresb` 残基B的新名称, 同上.

5.7 文件格式

5.7.1 拓扑文件

拓扑文件是根据GROMACS对分子拓扑的具体说明建立的. 可利用 `pdb2gmx` 程序生成 `*.top` 文件. 拓扑文件中所有可能的输入项都列于表 5.4和表 5.5中. 表中还列有: 所有参数的单位, 哪些相互作用可被微扰以用于自由能计算, `grompp` 可使用哪些键合相互作用生成排除, `grompp` 可将哪些键合相互作用转化为约束.

目录

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

表5.4: 拓扑(*.top)文件

参数					
相互作用类型	指令	原子数	函数类型	参数	自由能
必需	defaults			非键函数类型; 组合规则 ^{cr} ; 生成对(no/yes); fudge LJ(); fudge QQ()	
必需	atomtypes			原子类型; 质量m(u); 电荷q(e); 粒子类型; v ^{cr} ; w ^{cr}	
	bondtypes	(参考表 5.5 bonds 指令)			
	pairtypes	(参考表 5.5 pairs 指令)			
	angletypes	(参考表 5.5 angles 指令)			
	dihedraltypes *	(参考表 5.5 dihedrals 指令)			
	constrainttypes	(参考表 5.5 constraints 指令)			
LJ	nonbond_params	2	1	v ^{cr} ; w ^{cr}	
Buckingham	nonbond_params	2	2	a(kJ mol ⁻¹); b(nm ⁻¹); c6(kJ mol ⁻¹ nm ⁶)	
分子定义					
相互作用类型	指令	原子数	函数类型	参数	自由能
必需	moleculetype			分子名称; n _{ex} ^{nrexcl}	
必需	atoms	1		原子类型; 残基编号; 残基名称; 原子名称; 电荷组编号; 电荷q(e); 质量m(u)	类型, q, m
分子内相互作用和几何构型定义见表 5.5					
系统					
相互作用类型	指令	原子数	函数类型	参数	自由能
必需	system			系统名称	
必需	molecules			分子名称; 分子数	

- 原子数: 对每个指令需要的原子类型的数目
- 函数类型: 要选择的函数类型
- 自由能: 表示自由能计算时这种相互作用的哪些参数可以进行插值
- ^cr^: 组合规则确定LJ参数的类型, 参见5.3.2节
- ^*^: 对 dihedraltypes 可指定4个原子或两个内原子(对异常二面角则设定两个外原子)
- ^nrexcl^: 对非键相互作用, 排除相邻的n_{ex}个键
- 对于自由能计算, 在正常参数之后, 类型, 电荷q和质量m或无参数添加到同一行上的拓扑'B'(λ=1).

表 5.5: [moleculetype] 指令详解

相互作用名称	拓扑文件指令	原子数*	函数	参数顺序及其单位	用于自由
--------	--------	------	----	----------	------

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.5.1 残基数据库

5.5.2 残基构建单元数据库

5.5.3 原子重命名数据库

5.5.4 氢数据库

5.5.5 末端数据库

5.5.6 虚拟位点数据库

5.5.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

			类型 [†]		能计算 [‡]
键	<code>bonds</code> \$, ¶	2	1	$b_0(\text{nm}); k_b(\text{kJ mol}^{-1} \text{nm}^{-2})$	所有
G96键	<code>bonds</code> \$, ¶	2	2	$b_0(\text{nm}); k_b(\text{kJ mol}^{-1} \text{nm}^{-4})$	所有
Morse键	<code>bonds</code> \$, ¶	2	3	$b_0(\text{nm}); D(\text{kJ mol}^{-1}); \beta(\text{nm}^{-1})$	所有
立方键	<code>bonds</code> \$, ¶	2	4	$b_0(\text{nm}); C_{i=2,3}(\text{kJ mol}^{-1} \text{nm}^{-i})$	
连接	<code>bonds</code> \$	2	5		
简谐势	<code>bonds</code>	2	6	$b_0(\text{nm}); k_b(\text{kJ mol}^{-1} \text{nm}^{-2})$	所有
FENE键	<code>bonds</code> \$	2	7	$b_m(\text{nm}); k_b(\text{kJ mol}^{-1} \text{nm}^{-2})$	
表格键	<code>bonds</code> \$	2	8	表号(≥ 0); $k(\text{kJ mol}^{-1})$	k
表格键	<code>bonds</code>	2	9	表号(≥ 0); $k(\text{kJ mol}^{-1})$	k
约束势	<code>bonds</code>	2	10	下限, 上限1, 上限2(nm); $k_{dr}(\text{kJ mol}^{-1} \text{nm}^{-2})$	所有
额外LJ或库仑	<code>pairs</code>	2	1	$V^{**}; W^{**}$	所有
额外LJ或库仑	<code>pairs</code>	2	2	$\text{fudge } QQ(); q_i, q_j(\text{e}); V^{**}; W^{**}$	
额外LJ或库仑	<code>pairs_nb</code>	2	1	$q_i, q_j(\text{e}); V^{**}; W^{**}$	
键角	<code>angles</code> ¶	3	1	$\theta_0(\text{deg}); k_\theta(\text{kJ mol}^{-1} \text{rad}^{-2})$	所有
G96键角	<code>angles</code> ¶	3	2	$\theta_0(\text{deg}); k_\theta(\text{kJ mol}^{-1})$	所有
键-键交叉项	<code>angles</code>	3	3	$r_{1e}, r_{2e}(\text{nm}); k_{rr'}(\text{kJ mol}^{-1} \text{nm}^{-2})$	
键-角交叉项	<code>angles</code>	3	4	$r_{1e}, r_{2e}, r_{3e}(\text{nm}); k_{r\theta}(\text{kJ mol}^{-1} \text{nm}^{-2})$	
Urey-Bradley键角	<code>angles</code> ¶	3	5	$\theta_0(\text{deg}); k_\theta(\text{kJ mol}^{-1} \text{rad}^{-2}); r_{13}(\text{nm}); k_{UB}(\text{kJ mol}^{-1} \text{nm}^{-2})$	所有
四次键角	<code>angles</code> ¶	3	6	$\theta_0(\text{deg}); C_{i=0,1,2,3,4}(\text{kJ mol}^{-1} \text{rad}^{-i})$	
表格角	<code>angles</code>	3	8	表号(≥ 0); $k(\text{kJ mol}^{-1})$	k
限制弯曲势	<code>angles</code>	3	10	$\theta_0(\text{deg}); k_\theta(\text{kJ mol}^{-1})$	
正常二面角	<code>dihedrals</code>	4	1	$\phi_s(\text{deg}); k_\phi(\text{kJ mol}^{-1}); \text{多重数}$	ϕ, k
异常二面角	<code>dihedrals</code>	4	2	$\xi_0(\text{deg}); k_\xi(\text{kJ mol}^{-1} \text{rad}^{-2})$	所有
Ryckaert-Bellemans二面角	<code>dihedrals</code>	4	3	$C_0, C_1, C_2, C_3, C_4, C_5(\text{kJ mol}^{-1})$	所有
周期异常二面角	<code>dihedrals</code>	4	4	$\phi_s(\text{deg}); k_\phi(\text{kJ mol}^{-1}); \text{多重数}$	ϕ, k

目录

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

傅立叶二面角	dihedrals	4	5	C_0, C_1, C_2, C_3, C_4 (kJ mol ⁻¹)	所有
表格二面角	dihedrals	4	8	表号(≥ 0); k (kJ mol ⁻¹)	k
正常二面角 (多重)	dihedrals	4	9	ϕ_s (deg); k_ϕ (kJ mol ⁻¹); 多重数	ϕ, k
限制二面角	dihedrals	4	11	ϕ_0 (deg); k_ϕ (kJ mol ⁻¹)	
结合弯区扭转势	dihedrals	4	10	a_0, a_1, a_2, a_3, a_4 (kJ mol ⁻¹)	
排除	exclusions	1		一个或多个原子索引号	
约束	constraints ^s	2	1	b_0 (nm)	所有
约束	constraints	2	2	b_0 (nm)	所有
SETTLE	settles	1	1	d_{oh}, d_{HH} (nm)	
二体虚拟站点	virtual_sites2	3	1	$a()$	
三体虚拟站点	virtual_sites3	4	1	$a, b()$	
三体虚拟站点 (fd)	virtual_sites3	4	2	$a(); d$ (nm)	
三体虚拟站点 (fad)	virtual_sites3	4	3	θ (deg); d (nm)	
三体虚拟站点 (out)	virtual_sites3	4	4	$a, b(); c$ (nm ⁻¹)	
四体虚拟站点 (fdn)	virtual_sites4	5	2	$a, b(); c$ (nm)	
N体虚拟站点 (COG)	virtual_sitesn	1	1	一个或多个构建原子索引号	
N体虚拟站点 (COM)	virtual_sitesn	1	2	一个或多个构建原子索引号	
N体虚拟站点 (COW)	virtual_sitesn	1	3	一对或多对构建原子的索引号与权重	
位置约束	position_restraints	1	1	k_x, k_y, k_z (kJ mol ⁻¹ nm ⁻²)	所有
平底位置约束	position_restraints	1	2	g, r (nm), k (kJ mol ⁻¹ nm ⁻²)	
距离约束	distance_restraints	2	1	类型; 标签; 下限, 上限 1, 上限2 (nm); 权重()	
二面角约束	dihedral_restraints	4	1	ϕ_0 (deg); $\Delta\phi$ (deg);	所有
方向约束	orientation_restraints	2	1	exp.; 标签; α ; c (U nm ^{α}); obs. (U); 权重 (U ⁻¹)	
角度约束	angle_restraints	4	1	θ_0 (deg); k_c (kJ mol ⁻¹); 多重数	θ, k
角度约束 (z)	angle_restraints_z	2	1	θ_0 (deg); k_c (kJ mol ⁻¹); 多重数	θ, k

• *: 指令所需的原子数目

• †: 用于选择的函数类型编号

• ‡: 表示自由能计算时这种相互作用的哪些参数可以进行插值

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

- `$:` `grompp` 会利用这种相互作用类型产生排除
- `¶:` `grompp` 会将这种相互作用类型转换为约束
- `**:` 组合规则确定LJ参数的类型, 参见5.3.2节
- `||:` 无连接, 因此不会为这种相互作用产生排除

文件布局说明:

- 分号(`;`)和换行符之间为注释
- 忽略 `\` 后的换行符, 即 `\` 为续行符
- `[` 和 `]` 之间为指令
- 拓扑等级(必须遵循)包含三个层次:
 - 参数级别, 定义了一些力场的设定(见表 5.4)
 - 分子级别, 应包含一个或多个分子的定义(见表 5.5)
 - 体系级别, 只包含体系的特定信息(`[ system ]` 及 `[ molecules ]`)
- 每一项应用空格或制表符分格, 而不是逗号
- 分子中的原子应从1开始连续编号
- 相同电荷组中的原子必须连续列出
- 文件只解析一次, 这意味着不会处理向前引用: 每一项在使用前必须进行定义
- 可以从键生成排除, 或手动覆盖
- 可以从原子类型生成键合力的类型, 或对每条键覆盖
- 在同一个原子上可以使用多个相同类型的键合相互作用
- 强烈推荐使用描述性的注释行和空行
- 自GROMACS 3.1.3版本开始, 参数级别的所有指令可使用多次, 并且没有顺序限制, 除了在其它参数定义中, 原子类型在使用前需要进行定义
- 对相同的原子类型组合, 如果某个相互作用的参数被定义了多次, 会使用最后定义的参数; 自GROMACS 3.1.3版本开始, 参数若存在不同值的重定义, `grompp` 会生成警告
- 在 `[ moleculetype ]` 之前使用 `[ atoms ]`, `[ bonds ]`, `[ pairs ]`, `[ angles ]` 等指令毫无意义, 会导致警告
- 在 `[ system ]` 之前使用 `[ molecules ]` 指令毫无意义, 会导致警告
- 在 `[ system ]` 之后唯一可以使用的指令是 `[ molecules ]`
- 在 `[ ]` 中使用未知的字符串会导致直到下一个指令的所有数据被忽略, 并产生警告

下面是拓扑文件的一个例子, `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
    
```

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.5.1 残基数据库

5.5.2 残基构建单元数据库

5.5.3 原子重命名数据库

5.5.4 氢数据库

5.5.5 末端数据库

5.5.6 虚拟位点数据库

5.5.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

```

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 kfacs 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

```

下面是此文件的解释说明.

`#include "amber99.ff/forcefield.itp"` : 包含你正使用的力场的信息, 包括键和非键参数. 本例使用了AMBER99力场, 但你进行模拟时可使用不同的力场. `grompp` 会自动找到这个文件并复制粘贴它的内容. 你可以在 `share/top/amber99.ff/forcefield.itp` 文件中看到它的内容

```

#define _FF_AMBER
#define _FF_AMBER99

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

#include "ffnonbonded.itp"

```

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

```
#include "ffbonded.itp"
```

```
#include "gbsa.itp"
```

两个 `#define` 语句设置了条件, 这样拓扑文件后面的部分可以知道使用了AMBER 99力场.

[defaults] :

- `nbfunc` 为非键函数类型, 取1(Lennard-Jones)或2(Buckingham)
- `comb-rule` 为组合规则的编号(参见5.3.2节).
- `gen-pairs` 用于对的生成. 默认为'no', 即得到对类型列表的1-4的参数. 当列表中不存在参数时, 会导致致命错误并停止运行. 设置为'yes'会根据使用 `fudgeLJ` 的正常 Lennard-Jones参数, 生成不存在于配对列表中的1-4参数
- `fudgeLJ` 为Lennard-Jones 1-4相互作用的修正因子, 默认为1
- `fudgeQQ` 为静电1-4相互作用的修正因子, 默认为1
- `N`为6-N势中排斥项的次数(只用于非键Lennard-Jones), 自GROMACS 4.5版本开始, `mdrun` 会读取和使用`N`, 当`N`不等于12时, 使用表格相互作用函数(在旧版本中, 你必须使用用户自定义的表格相互作用).

注意, `gen-pairs`, `fudgeLJ`, `fudgeQQ` 和`N`是可选的. 仅当 `gen-pairs` 被设置为'yes'时才会使用 `fudgeLJ`, 而总会使用 `fudgeQQ`. 然而, 如果你想指定`N`, 你也需要给出其他参数的值.

后面的其他一些 `#include` 语句用于添加描述力场其余部分所需的大量数据. 我们将跳过这些并回到 `urea.top` 文件. 在那里, 我们会看到

[moleculetype] : 定义此 `*.top` 文件中分子的名称, `nrexcl=3`代表排除不远于三条键的原子之间的非键相互作用.

[atoms] : 定义分子, 其中 `nr` 和 `type` 是固定的, 其余的由用户自定义. 因此 `atom` 可以随意命名, `cgnr` 可大可小(如果可能, 一个电荷组的总电荷应为零), 这里的电荷也可以更改.

[bonds] : 不注释.

[pairs] : LJ和库仑的1-4相互作用

[angles] : 不注释.

[dihedrals] : 在此情况下, 有9个正常二面角(`funct=1`), 3个异常二面角(`funct=4`), 没有 Ryckaert-Bellemans型的二面角. 如果你想包含Ryckaert-Bellemans型的二面角, 请遵照下列的示例(例如, 对癸烷):

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

在烷烃势能的原始实现中[128], 没有使用1-4相互作用, 这意味着为了使用这个特定的力场, 你需要从拓扑文件的 [pairs] 段中移除1-4相互作用. 对大多数的现代力场, 如 OPLS/AA或Amber, 其规则是不同的, Ryckaert-Bellemans势是作为余弦序列与1-4相互作用一起使用的.

[position_restraints] : 利用简谐势将选中的粒子约束在参考位置(参见4.3.1节). `grompp` 会从由单独的坐标文件中读取参考位置.

[dihedral_restraints] : 将选定的二面角约束在参考值. 二面角约束的实现方法参见本手册4.3.4节的论述. [dihedral_restraints] 指令中设定的参数如下:

- `type` 只有一个可能的值, 1
- `label` 未使用, 并被从代码删除.
- `phi` 为本手册方程4.84和4.85中的 ϕ_0 值.

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

- `dphi` 为本手册方程4.85中的 $\Delta\phi$ 值.

- `kfac` 类似于距离约束中的 `fac`, 是力常数的修正因子. 通过它, 可以利用不同的力常数维持不同的约束.

- `power` 未使用, 并被从代码删除.

`#include "tip3p.itp"`: 包括一个已经构建好的拓扑文件(参见5.7.2节).

`[ system ]`: 体系的标题, 用户自定义

`[ molecules ]`: 定义体系中分子(亚分子)的总数, 这些分子已经在此 `*.top` 中进行了定义. 在这个示例文件中, 它代表1个尿素分子溶于1000个水分子中. 分子类型SOL是在 `tip3p.itp` 文件中定义的. 这里的每个名称必须对应于拓扑文件前面的 `[ moleculetype ]` 中给出的名称. 分子类型块的顺序, 这些分子的数目必须与坐标文件匹配, 坐标文件和拓扑文件会一起提供给 `grompp`. 分子块不需要连续, 但某些工具(如 `genion`)可能只对一个特定分子类型的第一或最后一块起作用. 另外, 这些块与组的定义无关(参见3.3节和8.1节).

5.7.2 Molecule.itp文件

如果你构建了一个拓扑文件并且经常使用(像水分子, 已经构建好的 `tip3p.itp`), 将它做成 `molecule.itp` 文件更好一些. 此文件中只列出了一个特定分子的信息, 你可以在多个体系中重用 `[ moleculetype ]`, 而无需重新调用 `pdb2gmx` 或手动复制粘贴. 例子 `urea.itp` 如下:

```
[ moleculetype ]
; molname nrexcl
URE 3

[ atoms ]
  1  C  1  URE    C    1  0.880229  12.01000  ; amber C type
...
  8  H  1  URE   H2    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
```

使用 `*.itp` 文件会使 `*.top` 文件明显变短:

```
;          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
```

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

5.7.3 ifdef语句

GROMACS有一个非常强大的功能, 你可以在 `*.top` 文件中使用 `#ifdef` 语句. 通过使用这条语句以及相关的 `#define` 语句, 像在前面的 `amber99.ff/forcefield.itp` 文件中那样, 在同一个 `*.top` 文件中可以为分子使用不同的参数. 下面给出TFE的一个例子, 其中的一个选项对原子使用了不同的电荷: De Loof等人给出的电荷[129]或Van Buuren和Berendsen给出的电荷[130]. 实际上, 你可以使用C预处理器 `cpp` 的许多功能, 因为 `grompp` 包含了类似的预处理函数以扫描文件. 使用 `#ifdef` 选项的方式如下:

- 在 `*.mdp` 文件(包含 `grompp` 输入参数)中使用选项 `define = -DDeLoof`, 或在 `*.top` 中早些使用 `#define DeLoof`, 并且

- 将 `#ifdef` 语句放在 `*.top` 中, 如所示:

```
...

[ 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
...
```

`pdb2gmx` 使用这个机制来实现可选的位置约束(4.3.1节), 具体是通过使用 `#include` 包含 `.itp` 文件, 而此文件的只有设置了特定的 `#define` (并且拼写正确!)时才有意义.

5.7.4 用于自由能计算的拓扑文件

两个体系A和B之间的自由能差值, 其计算方法见3.12节的论述. 描述体系A和B的拓扑具有相同数目的分子, 分子的原子数也相同. 通过在 `[ atoms ]` 指令下添加B参数, 可以对质量和非键相互作用进行微扰. 键合相互作用的微扰可以通过添加B参数到键合类型或键合相互作用来完成. 能够进行微扰的参数列于表 5.4和表 5.5. 相互作用的 λ 依赖性见4.5节的论述. 使用的键参数(位于定义键相互作用的行, 或通过对键合类型列表的原子类型进行查找得到)在表 5.6中进行了解释. 大多数情况下, 处理方式都很直观. 当键合相互作用中A和B的原子类型不完全相同, 且B状态的参数不存在时, 无论是位于行还是键合类型, `grompp` 都会使用A状态的参数, 并发出警告. 对自由能计算, 拓扑B($\lambda = 1$)的所有参数或者添加在同一行, 或者不添加, 添加时位于正常参数之后, 顺序与正常参数相同. 自GROMACS 4.6版本起, 如果 λ 被视为矢量, 那么 `bonded-lambdas` 分量控制所有未明确标记为约束的键合项. 约束项通过 `restraint-lambdas` 分量控制.

表 5.6: 用于自由能拓扑的键合参数, 位于定义键合相互作用的行, 或基于原子类型在键类型中查找. A和B表示分别用于状态A和B的参数, +和-表示拓扑中(不)存在的参数, x

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

表示存在没有影响。

B状态与A状态的原子类型完全相同?	行参数 A B	A原子类型的键合类型参数 A B	B原子类型的键合类型参数 A B	信息
yes	+AB - +B - - - -	x x x - - +AB - +A +B		错误
no	+AB - +B - - - - - -	x x x - - +AB - +A +B +A x	x x x x - - - +B - +B	警告 错误 警告 警告 警告

下面是一个拓扑文件的例子, 将200个丙醇分子转变为200个戊烷分子, 使用了GROMOS-96力场.

```
; Include force field parameters
#include "gromos43a1.ff/forcefield.itp"
[ moleculetype ]
; Name      nrexcl
PropPent    3

[ atoms ]
; nr      type  resnr  residue  atom  cgnr  charge  mass  typeB  char
1         H     1      PROP     PH     1     0.398   1.008  CH3     e
2         OA     1      PROP     PO     1    -0.548  15.9994 CH2     e
3         CH2    1      PROP     PC1    1     0.150   14.027  CH2     e
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
```

未微扰的 PC2 和 PC3 原子不需要指定B状态参数, 因为所用B参数将从A参数复制得来. 原子间不微扰的键合相互作用不需要指定B参数, 如示例拓扑中的最后一条键. 使用OPLS/AA

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

力场的拓扑根本不需要键合参数, 因为A和B参数都由原子类型决定. 涉及一个或两个微扰原子的非键相互作用将使用自由能微扰函数形式, 两个非微扰原子间的非键相互作用使用正常的函数形式. 这意味着, 例如, 当只对粒子的电荷进行微扰时, 若 λ 不等于0或1, 其Lennard-Jones相互作用也会受到影响.

注意, 此拓扑文件使用了GROMOS-96力场, 其中的键合相互作用不是由原子类型决定的. 键合相互作用的字符串由C预处理器转换而来. 力场参数文件包含这样的行:

```
#define gb_26 0.1530 7.1500e+06
```

```
#define gd_17 0.000 5.86 3
```

5.7.5 约束力

一个分子中两个原子之间的约束力, 可以使用自由能微扰代码进行计算, 计算时在两个原子之间添加约束, 约束对拓扑A和B具有不同长度. 当B的长度比A大1 nm时, λ 保持为零, 哈密顿量对 λ 的导数就是约束力. 对分子之间的约束, 可以使用拉扯代码, 参见6.4节. 下面是一个具体的例子, 通过组合两个甲烷成一个“分子”, 计算水中的两个甲烷分子距离0.7 nm时的约束力. **注意**, GROMACS中“分子”的定义不一定对应于分子的化学定义. 在GROMACS中, 一个“分子”可被定义为希望同时考虑的任意一组原子. 添加的约束函数类型为2, 意味着它不被用于生成排除(参见5.4节). 注意约束自由能项被包含在导数项中, 并且具体被包含在 **bonded-lambdas** 部分中. 然而, 改变约束的自由能 **没有** 被包含在用于BAR和MBAR的势能差值中, 因为这需要对每个约束分量重新计算能量. 我们计划在以后的版本中实现此功能.

```
; 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 坐标文件

扩展名为 **.gro** 的文件包含了GROMOS-87格式分子结构. 这种文件的一个示例片段如下:

```
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
```

格式是固定, 即, 所有列的位置都固定. 如果你想在自己的程序中读取这样的文件, 而不使用GROMACS库, 你可以使用以下格式:

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

C格式: `"%5i%5s%5s%5i%8.3f%8.3f%8.3f%8.4f%8.4f%8.4f"`

或更准确一点, 包含标题等内容, 应该像这样:

```
"%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%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格式: `(i5, 2a5, i5, 3f8.3, 3f8.4)`

因此 `conf.in.gro` 是GROMACS坐标文件, 几乎与GROMOS-87文件(对GROMOS用户: 当使用 `ntx=7` 时)相同. 唯一的区别是, GROMACS对盒子使用了张量, 而不是向量.

5.8 力场的组织

5.8.1 力场文件

默认提供了许多力场, 并根据 `<name>.ff` 目录的存在进行检测, 位于 `$GMXLIB/share/gromacs/top` 的子目录和/或工作目录中. `pdb2gmh` 会打印出力场文件的位置信息, 因此你可以很容易地知道调用了哪个版本的力场, 以防你在一个或另一个位置进行了修改. GROMACS包含了以下力场:

- AMBER03蛋白质, 核酸 AMBER94(Duan et al., *J. Comp. Chem.* 24, 1999–2012, **2003**)
- AMBER94力场(Cornell et al., *JACS* 117, 5179–5197, **1995**)
- AMBER96蛋白质, 核酸AMBER94(Kollman et al., *Acc. Chem. Res.* 29, 461–469, **1996**)
- AMBER99蛋白质, 核酸AMBER94((Wang et al., *J. Comp. Chem.* 21, 1049–1074, **2000**)
- AMBER99SB蛋白质, 核酸AMBER94(Hornak et al., *Proteins* 65, 712–725, **2006**)
- AMBER99SB-ILDN蛋白质, 核酸AMBER94(Lindorff-Larsen et al., *Proteins* 78, 1950–58, **2010**)
- AMBERGS力场(Garcia & Sanbonmatsu, *PNAS* 99, 2782–2787, **2002**)
- CHARMM27全原子力场(CHARM22与CMAP的蛋白质力场)
- GROMOS96 43a1力场
- GROMOS96 43a2力场(改进了烷烃二面角)
- GROMOS96 45a3力场((Schuler *JCC* **2001** 22 1205)
- GROMOS96 53a5力场(*JCC* **2004** vol 25 pag 1656)
- GROMOS96 53a6力场(*JCC* **2004** vol 25 pag 1656)
- GROMOS96 54a7力场(*Eur. Biophys. J.* (**2011**), 40, 843–856, DOI: 10.1007/s00249–011–0700–9)
- OPLS-AA/L全原子力场(2001氨基酸二面角)

力场被包含在拓扑文件的开始, 使用 `#include` 语句, 后面跟着

`<name>.ff/forcefield.itp`. 此语句包含力场文件, 反过来, 被包含的力场文件也可包含其它力场文件. 所有的力场以同样方式进行组织. 力场的一个例子 `amber99.ff/forcefield.itp` 见5.7.1节.

对每个力场, 有一些文件只被 `pdb2gmh` 使用. 它们是: 残基数据库(`.rtp`, 参见5.6.1节), 氢数据库(`.hdb`, 参见5.6.4节), 两个末端数据库(`.n.tdb` 和 `.c.tdb`, 参见5.6.5节), 只包含质量的原子类型数据库(`.atp`, 参见5.2.1节). 其他可选的文件参见5.6节的论述.

5.1 简介

5.2 粒子类型

5.2.1 原子类型

5.2.2 虚拟位点

5.3 参数文件

5.3.1 原子

5.3.2 非键参数

5.3.3 键参数

5.3.4 分子内的对相互作用

5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

5.6.1 残基数据库

5.6.2 残基构建单元数据库

5.6.3 原子重命名数据库

5.6.4 氢数据库

5.6.5 末端数据库

5.6.6 虚拟位点数据库

5.6.7 特殊键

5.7 文件格式

5.7.1 拓扑文件

5.7.2 Molecule.itp文件

5.7.3 ifdef语句

5.7.4 用于自由能计算的拓扑文件

5.7.5 约束力

5.7.6 坐标文件

5.8 力场的组织

5.8.1 力场文件

5.8.2 改变力场参数

5.8.3 添加原子类型

5.8.2 改变力场参数

如果你想更改分子中少量的键相互作用的参数, 最容易的方法是, 直接在 `*.top` 文件 `[ moleculetype ]` 段(参见5.7.1节的格式和单位说明)的键相互作用的定义之后输入参数. 如果想更改一种相互作用所有实例中的参数, 你可以在力场文件更改, 或者在包含力场之后添加一个新的 `[ ???types ]` 段. 如果一种相互作用参数被定义了多次, 会使用最后的定义. 对GROMACS 3.1.3版本, 当使用不同的值重定义参数时, 会导致警告. 建议不要更改原子类型的Lennard-Jones参数, 因为在GROMOS力场中一些原子类型组合的Lennard-Jones参数并不是根据标准组合规则生成的. 这些组合(以及其他可能遵循组合规则的)在 `[ nonbond_params ]` 段进行定义, 更改原子类型的Lennard-Jones参数对这些组合没有影响.

5.8.3 添加原子类型

对GROMACS 3.1.3版本, 在包含正常力场之后, 可以使用额外的 `[ atomtypes ]` 段添加原子类型. 在新的原子类型定义之后, 可以定义另外的非键参数和对参数. 对3.1.3版本以前的GROMACS, 新的原子类型需要添加到力场文件的 `[ atomtypes ]` 段, 因为在最后的 `[ atomtypes ]` 段之上的所有非键参数都会使用标准的组合规则覆盖.



◆本文地址: <https://jerkwin.github.io/GMX/GMXman-5/>, 转载请注明◆

◆评论问题: <https://jerkwin.herokuapp.com/category/3/博客>, 欢迎留言◆

5.1 简介

5.2 粒子类型

- 5.2.1 原子类型
- 5.2.2 虚拟位点

5.3 参数文件

- 5.3.1 原子
- 5.3.2 非键参数
- 5.5.3 键参数
- 5.3.4 分子内的对相互作用
- 5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

- 5.6.1 残基数据库
- 5.6.2 残基构建单元数据库
- 5.6.3 原子重命名数据库
- 5.6.4 氢数据库
- 5.6.5 末端数据库
- 5.6.6 虚拟位点数据库
- 5.6.7 特殊键

5.7 文件格式

- 5.7.1 拓扑文件
- 5.7.2 Molecule.itp文件
- 5.7.3 ifdef语句
- 5.7.4 用于自由能计算的拓扑文件
- 5.7.5 约束力
- 5.7.6 坐标文件

5.8 力场的组织

- 5.8.1 力场文件
- 5.8.2 改变力场参数
- 5.8.3 添加原子类型

5.1 简介

5.2 粒子类型

- 5.2.1 原子类型
- 5.2.2 虚拟位点

5.3 参数文件

- 5.3.1 原子
- 5.3.2 非键参数
- 5.5.3 键参数
- 5.3.4 分子内的对相互作用
- 5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

- 5.6.1 残基数据库
- 5.6.2 残基构建单元数据库
- 5.6.3 原子重命名数据库
- 5.6.4 氢数据库
- 5.6.5 末端数据库
- 5.6.6 虚拟位点数据库
- 5.6.7 特殊键

5.7 文件格式

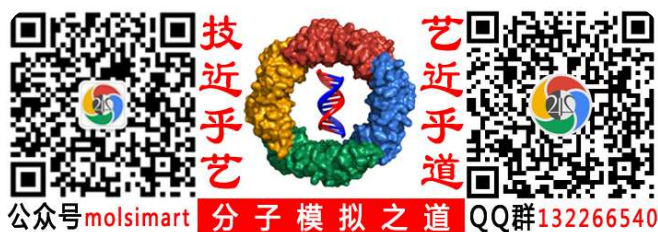
- 5.7.1 拓扑文件
- 5.7.2 Molecule.itp文件
- 5.7.3 ifdef语句
- 5.7.4 用于自由能计算的拓扑文件
- 5.7.5 约束力
- 5.7.6 坐标文件

5.8 力场的组织

- 5.8.1 力场文件
- 5.8.2 改变力场参数
- 5.8.3 添加原子类型

前一篇:

后一篇:



访问人次(2015年7月 9日起): 1011312 | 最后更新: 2022-01-06 21:55:48 UTC | 版权所有 © 2008 - 2022
Jerkwin

目录

5.1 简介

5.2 粒子类型

- 5.2.1 原子类型
- 5.2.2 虚拟位点

5.3 参数文件

- 5.3.1 原子
- 5.3.2 非键参数
- 5.3.3 键参数
- 5.3.4 分子内的对相互作用
- 5.3.5 隐式溶剂化模型

5.4 排除

5.5 约束算法

输入文件

- 5.6.1 残基数据库
- 5.6.2 残基构建单元数据库
- 5.6.3 原子重命名数据库
- 5.6.4 氢数据库
- 5.6.5 末端数据库
- 5.6.6 虚拟位点数据库
- 5.6.7 特殊键

5.7 文件格式

- 5.7.1 拓扑文件
- 5.7.2 Molecule.itp文件
- 5.7.3 ifdef语句
- 5.7.4 用于自由能计算的拓扑文件
- 5.7.5 约束力
- 5.7.6 坐标文件

5.8 力场的组织

- 5.8.1 力场文件
- 5.8.2 改变力场参数
- 5.8.3 添加原子类型