如何利用Ambertools联合Moltemplate构建GAFF/GAFF2势函数的标准lt模板文件

时间：2023/09/05

刘长伟

Y11190003@mail.ecust.edu.cn

准备工作

- 编译Ambertools工具
- 编译moltemplate

参考教程

- Recommended way to make LT files
- As of 2022-8-24, all of the LT files in these examples were created manually (by me). This is problematic because I had to obtain the atomic charges by other means and I had to guess the correct AMBER-specific @atom types for each atom in the molecule (eg “@atom:ca” in “benzene.lt”). This is not feasible for large, complex molecules. Instead, I suggest using AmberTools to create MOL2 and FRCMOD files that define your molecule. These files MOL2 and FRCMOD files will hopefully contain reasonable charges and @atom types for your molecule (although you should still check them before use). Then convert those files into moltemplate format automatically using amber2lt.py and mol22lt.py.
- moltemplate/examples/all_atom/force_field_AMBER at master ·jewettaij/moltemplate ·GitHub

编译Ambertools

- https://ambermd.org/GetAmber.php#ambertools

To install, proceed as follows:

- If you don't have conda installed, please visit the Miniconda download page.
- Those with an existing conda installation may wish to create a new conda "environment" to avoid conflicts with what you already have installed. To do this:

```
conda create --name AmberTools23
conda activate AmberTools23
```

(Note that you would need to perform the "conda activate" step every time you wish use AmberTools23 in a new terminal; it might be appropriate to add this to your start-up script. Creating a new environment should not be necessary if you only use conda for AmberTools.)

- Once this is done, type:

```
conda install -c conda-forge ambertools=23
```

- AmberTools is updated from time to time. To keep your conda package up-to-date, do this:

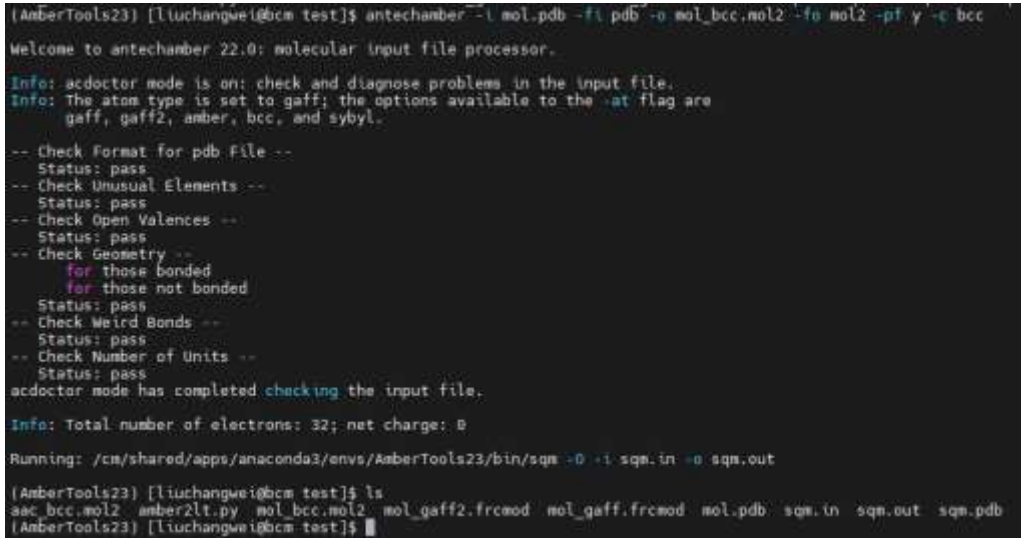
```
conda update -c conda-forge ambertools
```

Thanks to Jaime Rodríguez-Guerra for spearheading this.

1，必须联网
2，在root账户下操作
3，ssh root@172.20.181.237
4，df –h
5， cd /cm/shared/apps/
6， mkdir ambertools
7，cd ambertools/
8，ping www.baidu.com
9，conda env list
10，firefox联网
11，conda create --name AmberTools23
12，conda activate AmberTools23
13，conda install -c conda-forge ambertools=23
14，在自己的账号下进入AmberTools23环境下即可操作了

使用antechamber处理得到mol_bcc.mol2文件

- 准备pdb文件或者mol2分子结构文件
- 运行：
- *antechamber -i mol.pdb -fi pdb -o mol_bcc.mol2 -fo mol2 -pf y -c bcc*
- *antechamber -i mol.mol2 -fi pdb -o mol_bcc.mol2 -fo mol2 -pf y -c bcc*
- -c选项则可以指定电荷计算的方法，-cf可以指定电荷的文件而不是进行计算．一般使用bcc电荷就好了(am1-bcc), 也可以使用基于量化计算结果的ESP/RESP电荷。
- -i, -fi和-o, -fo 分别是输入文件, 输入文件类型, 输出文件, 输出文件类型.
- -pf y 可以忽略计算中产生的中间文件, 墙裂建议!



- 参考：
- https://gohom.win/2015/11/12/AMBER-Ligand/#fn:CalcRESP

使用parmchk2处理得到mol_gaff.frcmod文件

- parmchk2是检查小分子结构在GAFF(默认, 也可以自己指定力场,参见选项)下有哪些缺失参数. 找出参数并计算添加相应合适的缺失力场参数. 运行以下命令获得frcmod文件:
- *parmchk2 -i mol_bcc.mol2 -f mol2 -o mol_gaff2.frcmod -s 2 -a Y -pf 2 -FC 2*
- parmchk2是原先parmchk的增强版, 可以检查输入分子构型中GAFF的缺失参数, 并生成相应的补充参数文件Lig.frcmod
- frcmod用于提供补充的参数，原本力场里就有的就没给出
- 一开始运行*parmchk2 -i acc_bcc.mol2 -f mol2 -o mol.frcmod -s 1*
- *发现生成的mol.frcmod文件中什么参数都没有，就像这样*
- *后来使用parmchk2 –help发现需要加上-a Y关键词才可以，最终这样*
- ***-s 1是gaff力场， -s 2是gaff2力场***
- Frcmod中的格式可以参考: 这里有所有文件格式的说明

  http://ambermd.org/FileFormats.php

acc.frcmod

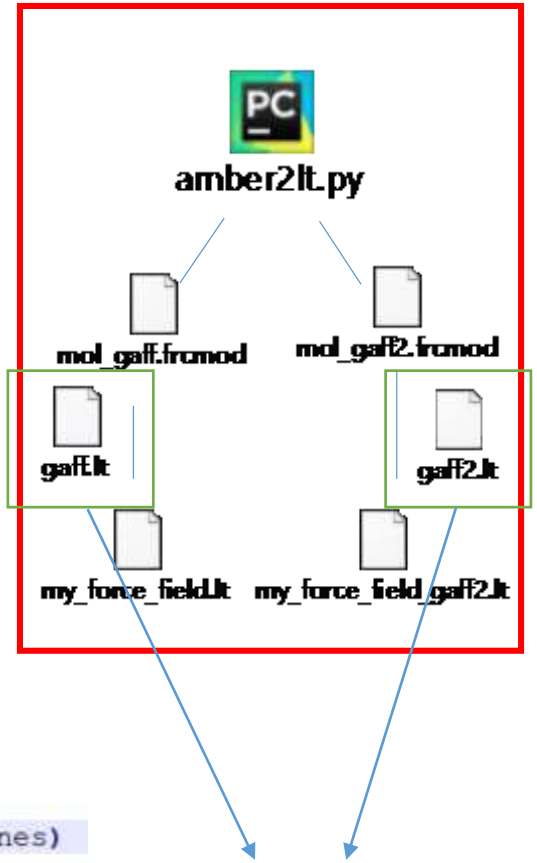
```
Remark line goes here
MASS

BOND

ANGLE

DIHE

IMPROPER

NONBON

```

ANTECHAMBER.FRCM... | mol.frcmod | mol.frcmod

```
Remark line goes here
MASS
c3 12.010        0.878
c  12.010        0.616
oh 16.000        0.465
hc 1.008         0.135
o  16.000        0.434
ho 1.008         0.135

BOND
c -c3  313.00   1.524
c3-hc  330.60   1.097
c -oh  400.10   1.351
c -o   637.70   1.218
ho-oh  371.40   0.973

ANGLE
c3-c -oh    68.400     112.730
c3-c -o     67.400     123.200
c -c3-hc    46.900     108.770
c -oh-ho    49.900     106.550
o -c -oh    75.900     122.100
hc-c3-hc    39.400     107.580

DIHE
c3-c -oh-ho   2    4.600        180.000          2.000
oh-c -c3-hc   6    0.000        180.000          2.000
o -c -c3-hc   1    0.800          0.000         -1.000
o -c -c3-hc   1    0.000          0.000         -2.000
o -c -c3-hc   1    0.080        180.000          3.000
o -c -oh-ho   1    2.300        180.000         -2.000
o -c -oh-ho   1    1.900          0.000          1.000

IMPROPER
c3-o -c -oh         1.1          180.0        2.0

NONBON
  c3           1.9080  0.1094
  c            1.9080  0.0860
  oh           1.7210  0.2104
  hc           1.4870  0.0157
  o            1.6612  0.2100
  ho           0.0000  0.0000
```

使用amber2lt.py生成force_filed.lt文件

- 将*amber2lt.py*和*mol_gaff.frcmod*放置在同一目录下面；
- 打开wsl
- 执行：
- *amber2lt.py --in mol.frcmod --name MyForceField >> my_force_field.lt*
- 生成my_force_field.lt
- 我发现使用amber2lt.py可以匹配出不同原子类型间的pair相互作用参数，这些参数从何而来？通过对比分析，发现这些参数来自于gaff.lt或者gaff2.lt力场文件。而键，角，二面角，非正常二面角的参数都来自frcmod文件中。
- Amber2lt.py如何识别gaff和gaff2力场呢，因为观察两个frcmod文件的内容，我们发现最后一组参数NONBON可以作为参考，因为NONBON这一项在Amber2lt.py中这样的定义



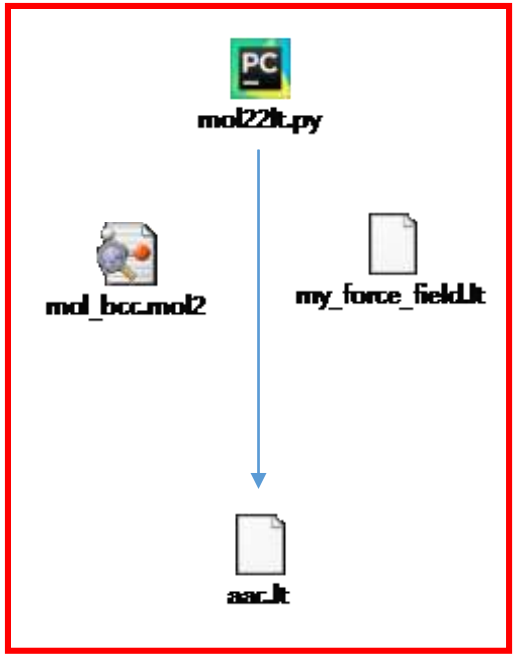
这两个立场文件不需要放在同一个目录下，因为amber2lt.py会调用moltemplate.sh程序，从而自动调用这两个文件。

```
def ExtractPairTextAB(lines, num_skip_lines = 0):
    return ExtractFFTableAB(lines, 1, ('NONB', 'NONBON', 'NONBOND', 'NONBONDED', 'MOD4'), num_skip_lines)
```

```
pair_coeff @atom:c3 @atom:c3 lj/charmm/coul/long 0.1094 3.3996695084235347
pair_coeff @atom:c @atom:c lj/charmm/coul/long 0.0860 3.3996695084235347
pair_coeff @atom:oh @atom:oh lj/charmm/coul/long 0.2104 3.0664733878390478
pair_coeff @atom:hc @atom:hc lj/charmm/coul/long 0.0157 2.649532787749369
pair_coeff @atom:o @atom:o lj/charmm/coul/long 0.2100 2.959921901149463
pair_coeff @atom:ho @atom:ho lj/charmm/coul/long 0.0000 0.0
```

最后使用mol22lt.py.来生成目标分子的lt模板文件

- *mol22lt.py --in mol_bcc.mol2 --out aac.lt --name Aac --ff MyForceField --ff-file my_force_field.lt*
- 最终得到lt模板。




```
import "my_force_field.lt"


Aac inherits MyForceField {

  #  atomId molId atomType charge X Y Z

  write("Data Atoms"){
      $atom:C $mol:m @atom:c3 -0.2001 1.374 -0.04 -0.0
      $atom:C1 $mol:m @atom:c 0.6231 -0.133 0.138 -0.0
      $atom:O $mol:m @atom:oh -0.5841 -0.855 -1.019 0.0
      $atom:H $mol:m @atom:hc 0.0747 1.85 0.939 -0.002
      $atom:H1 $mol:m @atom:hc 0.0747 1.697 -0.601 -0.884
      $atom:H2 $mol:m @atom:hc 0.0747 1.696 -0.597 0.887
      $atom:O1 $mol:m @atom:o -0.485 -0.698 1.2 0.0
      $atom:H3 $mol:m @atom:ho 0.422 -0.261 -1.783 -0.0
  } # Atoms section
```

Amber和Gaussian联用计算RESP原子电荷

- am1-bcc和resp电荷计算方法在很多计算方法中有一定的优势，但是读者切不可盲目，应根据自己研究的实际情况来选择原子电荷计算方法，否则得不偿失。
- **1、构建分子的结构文件，并存为.mol2文件**
- **2、采用gaussian优化结构**

```
关键词如下：
#p HF/6-31G* SCF Pop=MK iop(6/33=2) iop(6/42=6) iop(6/50=1) opt
在坐标的末尾输入两行，分别是“ligand_ini.gesp”和“ligand.gesp”，保存为ligand.gjf文件
```

- **3、采用amber拟合resp电荷**

antechamber -i ligand.out -fi gout -o ligand_resp.mol2 -fo mol2 -pf y -c resp

计算完成后得到ligand_resp.mol2文件中即包含了所有原子的resp电荷。