Small Molecule Dihedrals Parametrization | Barril Lab

Small Molecule Dihedrals Parametrization

Posted by Sergio Ruiz

Original entry published in CBDD Research Group Blog.

We have found ourselves several times in a situation where we have to parametrize compounds for running MDs.
In this blog entry, I will try to set up a protocol for parametrization of small molecules dihedrals for AMBER using Gaussian and sander (based on the tutorial found in this webpage).

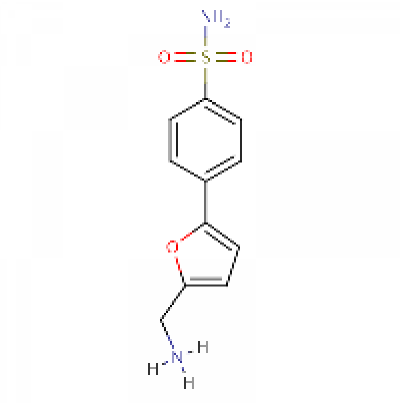
All files can be downloaded from this shared folder.

If you want, you can download the excel file in the folder (energies_benzfuran.xls), which could be useful in organizing all partial results and automatic calculations and plots to check all the process.

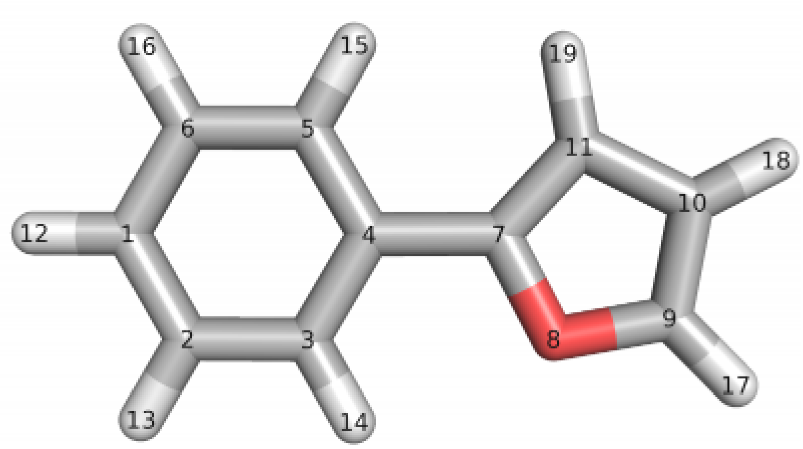
Overview:

- **Calculate reference with Gaussian.**
- **Run sander with values ELE+VdW (without dihedral contribution).**
- **Use a genetic algorithm to optimize dihedral values with respect to Gaussian reference.**
- **Modify frcmod parameters with the ones found in step before.**
- **Check with sander that the dihedral has the same profile as in Gaussian.**

The molecule I want to parametrize is the following:



I have selected as an example the dihedral between benzene and furan rings, and the structure I will use to do so is the following:



As it can be seen, I have numbered the atoms starting from one end of the molecule, the benzene, followed by the rest of benzene atoms. Then, the furan ring and, finally, the hydrogen atoms following a similar order as the heavy atoms.

```
babel -imol2 <MOL>.mol2 -opdb <MOL>.pdb
```

```
[edit <MOL>.pdb to correctly numerate atoms]
babel -ipdb <MOL>.pdb -omol2 <MOL>_correct.mol2
```

This has to be done to facilitate the creation of gaussian input files, which must be in internal coordinates format.

- `antechamber -i <MOL>_correct.mol2 -fi mol2 -o <MOL>.com -fo gzmat`

In this case, I want to parametrize the dihedral angle between both rings, defined by atoms 3-4-7-8.
So, we will have to check that there is only one defined torsion for the selected dihedral. We will edit <MOL>.com to ensure it is OK (in our case, the dihedral angle will be defined by t8 parameter, see below).

Gaussian

Get reference energy profile:

Gaussian will be used to obtain the energy profile of this dihedral that will act as a reference.
To do so, we will rotate the dihedral from 0 to 360 degrees (with a step size of 5 degrees) and minimize the rest of the structure.

The input file for this job looks as follows (benz_furan.com):

```
-Link1-

#HF/6-31G* SCF=tight Opt=ModRedundant



remark line goes here



0   1

    C

    C    1      b2

    [...]

    H    11     b19   10     a19    9     d19
    H   11     b19   10     a19    9     t19


b2=   1.3840

b3=   1.3830

a3= 119.5982

[…]

t19= -178.0170

t8=    0.000



8 7 4 3 0.0 S 72 5.0
```

With the important things being:

- `Opt=ModRedundant` for allowing the dihedral angle scan to be calculated.
- `t8=0`. This is the dihedral angle we want to scan and it has been initialized at 0 degrees.
- `8 7 4 3 0.0 S 72 5.0`. The dihedral we want to parametrize is defined by these atoms (8-7-4-3), the starting angle is 0.0 and we want it to scan (S) for 72 steps with a step-size of 5 degrees (from 0 to 360).

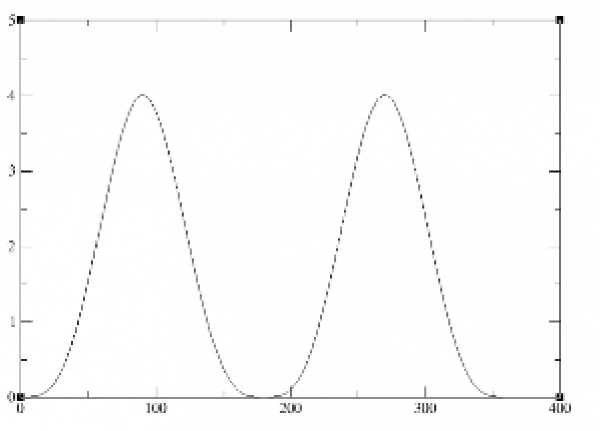
After running in gaussian:

- `g09 benz_furan.com benz_furan.log`

The output file (benz_furan.log) will contain a final section, from which the results that we will use can be extracted:

```
[…]
Version=AM64L-G09RevA.02\State=1-A\HF=-458.1826707,-458.1826681,-4

 58.1826538,-458.1826111,-458.1825183,-458.1823552,-458.1821061,-458.18

 17627,-458.1813241,-458.1807973,-458.1801971,-458.1795449,-458.1788687

 ,-458.1782015,-458.17758,-458.1770426,-458.1766263,-458.1763621,-458.1

 762706,-458.1763582,-458.1766159,-458.177022,-458.177546,-458.1781531,

 -458.1788075,-458.1794751,-458.1801245,-458.1807283,-458.1812642,-458.

 1817157,-458.1820733,-458.1823353,-458.1825083,-458.1826072,-458.18265

 29,-458.182668,-458.1826708,-458.182668,-458.1826528,-458.1826072,-458

 .1825083,-458.1823353,-458.1820733,-458.1817157,-458.1812642,-458.1807

 283,-458.1801245,-458.1794751,-458.1788075,-458.1781531,-458.177546,-4

 58.1770221,-458.1766159,-458.1763582,-458.1762706,-458.1763621,-458.17

 66263,-458.1770426,-458.17758,-458.1782015,-458.1788687,-458.1795449,-

 458.1801971,-458.1807973,-458.181324,-458.1817627,-458.1821061,-458.18

 23552,-458.1825183,-458.1826111,-458.1826539,-458.1826682,-458.1826708

 \RMSD=5.416e-09
[…]
```

The highlighted text above, from “HF=” to “\RMSD=” contains the energies calculated by Gaussian for all the 73 steps (72 increments plus first starting 0.0 degrees). These values units are Hartrees, which must be converted to kcal/mol by multiplying by 627.51 (1Hartree = 627.51 kcal/mol). Finally, we will calculate the minimum value and substract this value to the rest to obtain relative energies, which can be plotted to visualize the reference profile (y-axis with energy in kcal/mol and x-axis with dihedral angle degrees):



Sander

1.- Get energy profile without any dihedral parameter

Geometry optimization and resp charges calculation for sander input:

After finishing Gaussian section, we will run sander to compare the results with the reference profile and optimize parameters to fit the data as best as possible.

First of all, we need to optimize and calculate resp charges for input structure. This will also be done with Gaussian:

- `antechamber -i <MOL>.mol2 -fi mol2 -o <MOL>_opt.com -fo gcrt`

Which will look like as follows, (benz_furan_opt.com), (note that we changed calculations keys -as charge calculation will be done in following steps. Here it will only optimize geometry):

```
–Link1–

%chk=benzfuran

#P B3LYP/6-31G* opt


remark line goes here


0   1

   C    -3.2160      -0.0700       0.0080

   C    -2.5920       1.1660      -0.0010
[…]
```

To run it:

- `g09 <MOL>_opt.com <MOL>_opt.log`

Then, we can convert the log output (benz_furan_opt.log) to mol2 (benz_furan_opt.mol2) to view the geometry in any molecule viewer:

-
  ```
  antechamber -i <MOL>_opt.log -fi gout -o <MOL>_opt.mol2 -fo mol2
  ```

And, afterwards, we will use this optimized geometry to calculate the corresponding resp charges:

-
  ```
  antechamber -i <MOL>_opt.mol2 -fi mol2 -o <MOL>_opt_charges.com -fo gcrt
  ```

We will change the keys in this file, (benz_furan_opt_charges.com), to only calculate charges, without geometry optimization:

```
-Link1-

%chk=molecule

#P HF/6-31G* SCF=tight Pop=MK iop(6/33=2) iop(6/42=6)


remark line goes here


0   1

    C     -3.2520         -0.0650          0.0000

    C     -2.6030          1.1720         -0.0000

    C     -1.2120          1.2370         -0.0000

[…]
```

Then, run this job with Gaussian:

-
  ```
  g09 <MOL>_opt_charges.com <MOL>_opt_charges.log
  ```

When it has finished, we will convert the log output file (benz_furan_opt_charges.log) to mol2 (benz_furan_opt_charges.mol2) format, for example, and we will have the input for sander calculations (optimized geometry and with resp charges). Note here that flags have been added: “-at” to change atom typing to amber (if not writte, it will use gaff) and “-c” for changing charge type to resp:

-
  ```
  antechamber -i <MOL>_opt_charges.log -fi gout -o <MOL>_opt_charges.mol2 -fo mol2 -at amber -c resp
  ```

Final step to have a correct input structure file is to check the assigned atom types and change them, if desired. In our case, we changed a few of them in order to facilitate MD simulations (we had previously assigned them considering the full molecule. In this case, the substructure selected for parametrization will have the same atom types).

Note: In our case, we had an error afterwards because the optimized geometry from Gaussian had all atoms in the same coordinate plane and -I don’t know why- it rised an error when running sander. So, before going on, we used MOE (or any tool you like) to rotate a little (less than 1 degree) one of the rings in order to avoid having all atoms in same coordinate plane.

FRCMOD file creation and parameter definition

Once we have the input mol2 file in a correct format (benz_furan_opt_charges.mol2), we will use parmchk to check the atoms and atom types in our molecule and whether there are any missing parameters in the chosen forcefield (we will use amber parm99) or not (there will be).

- `parmchk -i <MOL>_opt_charges.mol2 -f mol2 -o <MOL>.frcmod -p $AMBERHOME/dat/leap/parm/parm99.dat`

Edit all parameters in frcmod file (benz_furan.frcmod) at your own criteria and leave the parameters (PK and PHASE) for the dihedral being analyzed in 0:

```
remark goes here

MASS


BOND

[…]


ANGLE

[…]


DIHE

X -CA-CC-X    4    0.000          0.000             1.000      ATTN, need revision

X -CC-C*-X    4   10.000        180.0               2.000

X -C*-C*-X    4    6.000        180.0               2.000

X -OS-CC-X    2   21.000        180.0               2.000


IMPROPER

[…]
```

NONBON

Sander simulation with dihedral energy set to 0

This way, setting the parameters to 0, we are performing with sander a minimization of the structure only taking into account Van der Waals and electrostatic contributions, without dihedral energies (only in the dihedral we are parametrizing).

To run the simulation, we will need to have in the folder the following files:

- `benz_furan.mol2` –-> (benz_opt_charges.mol2 with the name changed to benz_furan.mol2 to facilitate scripts running) from optimization and charge calculation (correct plane coordinates in our case).
- `benz_furan.frcmod` –> as defined above (dihedral defined to 0).
- `run_all_sander_param.sh` –> script that will create all needed dihedral constraints files, will run sander minimisations for all the dihedral angles, will extract the values of the energies calculated and will also convert all steps to sd format in order to view all the structures in each step in any molecular viewer.

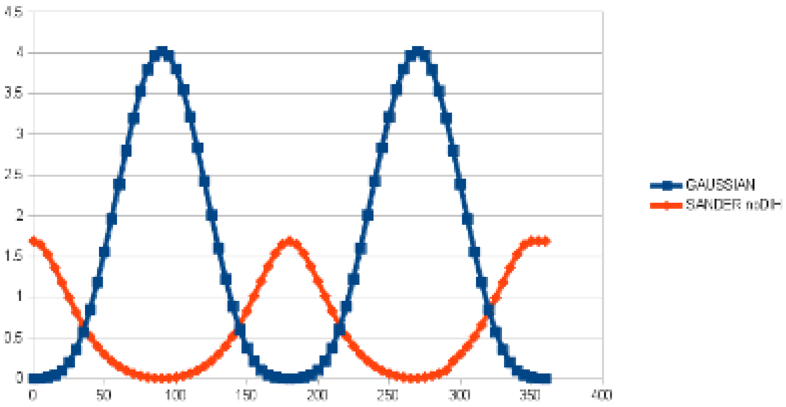
And, to execute it, just type:

- `sh run_all_sander_param.sh 3 4 7 8 benz_furan`
  - where `3 4 7 8` are the atoms forming the dihedral
  - `benz_furan` is the root name for all files

After finishing, some files and folder will appear in your directory:

- `benz_furan.tleap` –> tleap job file for processing frcmod and mol2 files
- `leap.log` –> log of the tleap process. It is important to check that charge is correctly defined and no errors have arised.
- `benz_furan.top benz_furan.off benz_furan.crd` –> files from tleap for running sander.
- `min.in` –> minimisation input template.
- `sander_energies.tab` –> energies for each step in a tabular format.
- `dih_tmp/` –> folder with the dihedral restraints files for each angle.
- `out_<ROOTNAME>/` –> folder with the output *out and *crd files from sander.
- `min_pdbs/` –> folder with the converted files to be viewed in a molecular viewer (for instance, "pymol min_pdbs/min_pdb_all.sd")

Then, the script will open using "kwrite" a file with the 73 corresponding energies of all the dihedrals (if "kwrite" is not found in the computer, an error will rise. Just open file "sander_energies.tab" with any text viewer of your choice).
The same process as in Gaussian profile generation will be applied here (except energies conversion from Hartrees to Kcal/mol) –> calculate minimum energy and substract to each step to get relative energies.
If we plot these energies compared with Gaussian:



Therefore, in the next steps, we will use a genetic algorithm to optimize the dihedral parameters in frcmod file to get a sander profile the most similar to the Gaussian profile as possible.

2.- Use a genetic algorithm to optimize dihedral values with respect to Gaussian reference

In this section, we will optimize the dihedral values found in frcmod file using a genetic algorithm from python package "pyevolve" (you will have to download this package to run this script — running "sudo easy_install pyevolve" has worked for us):

- GA_minimum.py

To understand what it does, taking a view to this page could be really helpful. The main issue here is to know that the formula applied to calculate the dihedral energy is the following:

Etors = (PK / IDIVF) * (1 + cos(PN * phi – PHASE))

Where PK, IDIVF, PN and PHASE are the values found in frcmod file:

DIHE
X -CA-CC-X IDIVF PK PHASE PN

To sum up, it will create a population with random values of this parameters, will let it evolve for 5000 generations and will print out the values of all the parameters for the best chromosome.

So, as we know the values of the energy without the contribution of this “Etors” value, by selecting the values of these paremeters that make the sum of “Etors” and the -previously calculated in sander- VdW and ELE terms the most similar to the Gaussian reference, we will obtain the optimal values for IDIVF, PK, PHASE and PN.

To run this script, we will need one file with the energies of Gaussian reference (the relative ones, with the minimum substracted) and another with sander energies (same as for Gaussian, relative energies). Then, type (the two tab files are also available in the shared folder):

- `python GA_minimum.py gaussian_relative_energies.tab sander_relative_energies.tab`

And the output, in our case:

```
Gen. 1 (0.02%): Max/Min/Avg Fitness(Raw) [0.09(0.43)/0.07(0.02)/0.08(0.08)]

Gen. 1000 (20.00%): Max/Min/Avg Fitness(Raw) [5.43(7.30)/3.06(0.03)/4.53(4.53)]

Gen. 2000 (40.00%): Max/Min/Avg Fitness(Raw) [9.46(9.46)/0.00(0.04)/8.70(8.70)]

Gen. 3000 (60.00%): Max/Min/Avg Fitness(Raw) [9.46(9.46)/0.00(0.03)/8.24(8.24)]

Gen. 4000 (80.00%): Max/Min/Avg Fitness(Raw) [9.47(9.46)/0.00(0.06)/8.51(8.51)]

Gen. 5000 (100.00%): Max/Min/Avg Fitness(Raw) [9.47(9.46)/0.00(0.06)/8.56(8.56)]

Total time elapsed: 321.672 seconds.

The optimized values for 2 functions are:

Function 1:

        PK/IDIVF =      2.8

        PHASE =         180.0

        PN =            2.0

Function 2:

        PK/IDIVF =      0.2

        PHASE =         0.0

        PN =            4.0
```

Hence, the values that are printed above are the values that we should add to the frcmod file to have a correct parametrization of the dihedral.

Note: In this case, as there are two functions we will have to add in the frcmod file a line in the way as follows (the “-” sign in front of PN means that there is another line that also defines the same dihedral. It is used when more than one function is needed to define the energy of a dihedral):

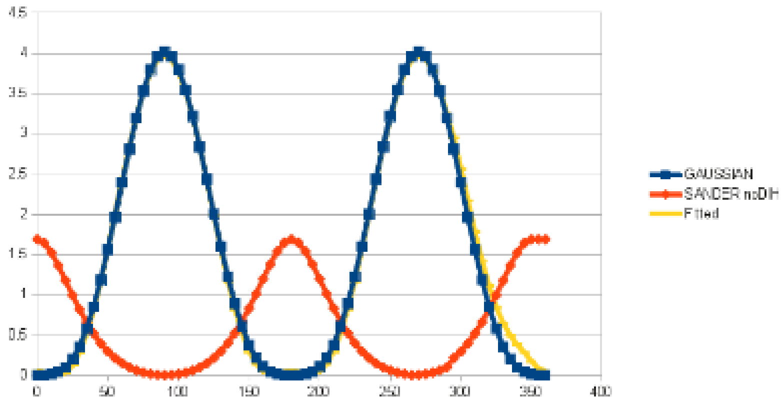
DIHE

X -CA-CC-X IDIVF1 PK1 PHASE1 **-PN1**

X -CA-CC-X IDIVF2 PK2 PHASE2 **PN2**

Note 2: (Thanks to Scott E Allen) Be careful when mixing dihedral wildcards (X-….-X) and odd PN values, as they could cancel each other out when applying the GA results to the frcmod file. A possible solution here is to split the wildcard dihedral into the different possible dihedral combinations.

If we plot the fitted theoretical profile of the energy calculated above adding the contributions of the two functions optimized, we will get this plot:



3.- Modify frcmod file parameters with the ones found in step before

In this section, we will have to edit the frcmod file used in steps before and add the two lines got in the genetic algorithm optimization.
The frcmod file will be as follows:

```
remark goes here

MASS


BOND

[…]


ANGLE

[…]


DIHE

X -CA-CC-X     4    2.800       180.0             -2.000

X -CA-CC-X     4    0.200         0.0              4.000

X -CC-C*-X     4   10.000       180.0              2.000

X -C*-C*-X     4    6.000       180.0              2.000

X -OS-CC-X     2   21.000       180.0              2.000


IMPROPER

[…]
```

> *NONBON*

Please note here the “trick” between the PK/IDIVF calculated in the genetic algorithm and the one that is actually written in frcmod file. As the formula applied in the optimization “thinks” that there is only one possible dihedral combination (in our case, X-CA-CC-X represents 4 different combinations) the value of PK/IDIVF is considering only 1 dihedral. I don’t really know how to make me more clear, so I encourage you to deepen in this issue to get a better understanding.
So, what we have to do is to modify the IDIVF value to the one defined by our system (in our case 4 because there are 4 different combinations) and write the predicted PK/IDIVF value in genetic algorithm in the PK spot in the frcmod file.

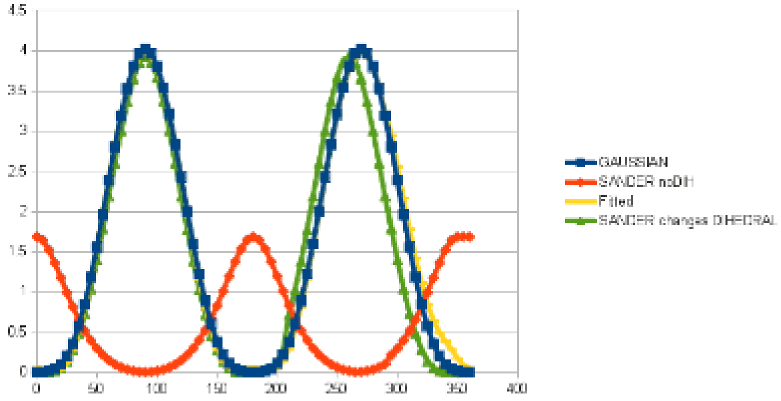
Once this file is correctly updated (benz_furan_CORRECTED.frcmod), we can run again “run_all_sander_param.sh” the same as in the previous section (same mol2 file but updated frcmod file) and check that the resulting energy profile is indeed similar to the Gaussian reference.

4.- Check with sander that the dihedral has the same profile as in Gaussian

Just run again the script in a new folder containing the mol2 and the corrected benz_furan.frcmod file (be sure it is called like this, with the same rootname as always):

- `sh run_all_sander_param.sh 3 4 7 8 benz_furan`

The results will also be opened in a “kwrite” window. Copy all energies found in “sander_energies.tab” and plot them comparing with the previously calculated (gaussian, sander without dihedral and theoretically fitted):



And, as it can be seen, the sander profile with the changes in frcmod (green line) is almost equal to Gaussian and fitted ones (blue and yellow, respectively).

The last step should be to update the original molecule frcmod file with the values calculated in all this process and it should work!
I hope it is clear enough and you find this little tutorial as useful as I found it myself, otherwise don’t hesitate to contact me if you need anything!

Sergio Ruiz

□ Blog