NCAP Tutorial

In this tutorial we will use NCAP to generate parameter and topology files for 5-Fluoro-L-tryptophan and model the mutation in a Trp-cage peptide. A Ubuntu workstation is recommended to complete this tutorial. The tools/dependencies you will need include:

- NCAP Binary - Latest Release
- NWChem - Ubuntu/Debian [`sudo apt install nwchem`]
- ChargeMol - `https://sourceforge.net/projects/ddec/files/`
- XTB - `https://github.com/grimme-lab/xtb`
- VMD - `https://www.ks.uiuc.edu/Research/vmd/`
- NAMD - `https://www.ks.uiuc.edu/Research/namd/`
- Conda/Mamba

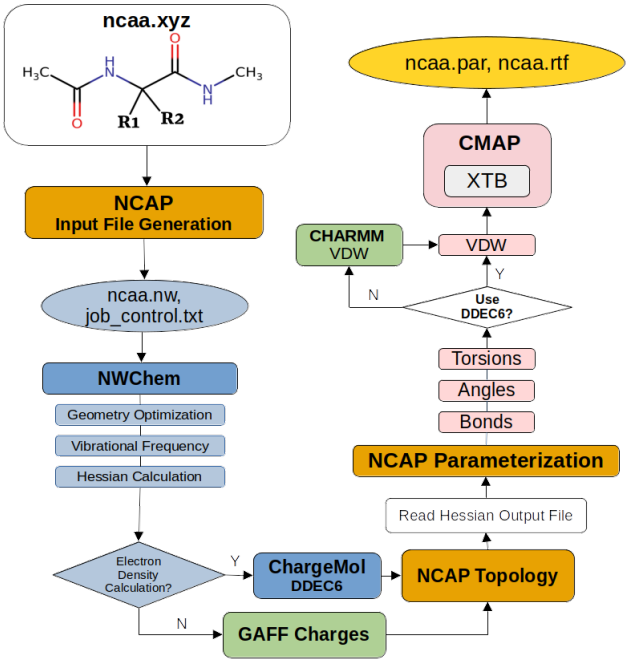

Figure 1: NCAP Workflow.

To start the tutorial run the following commands in a terminal.

```
$ cd example
$ cp 5-ftrp_input/ncaa.xyz 5ftrp.xyz
$ ncap 5ftrp.xyz
$ cd 5ftrp
```

At this stage NCAP will create a working directory base on the prefix of the file name and generate input files for NWChem and Chargemol. Once you change directories into the new `5ftrp` working directory you will be ready to run your NWChem calculation.

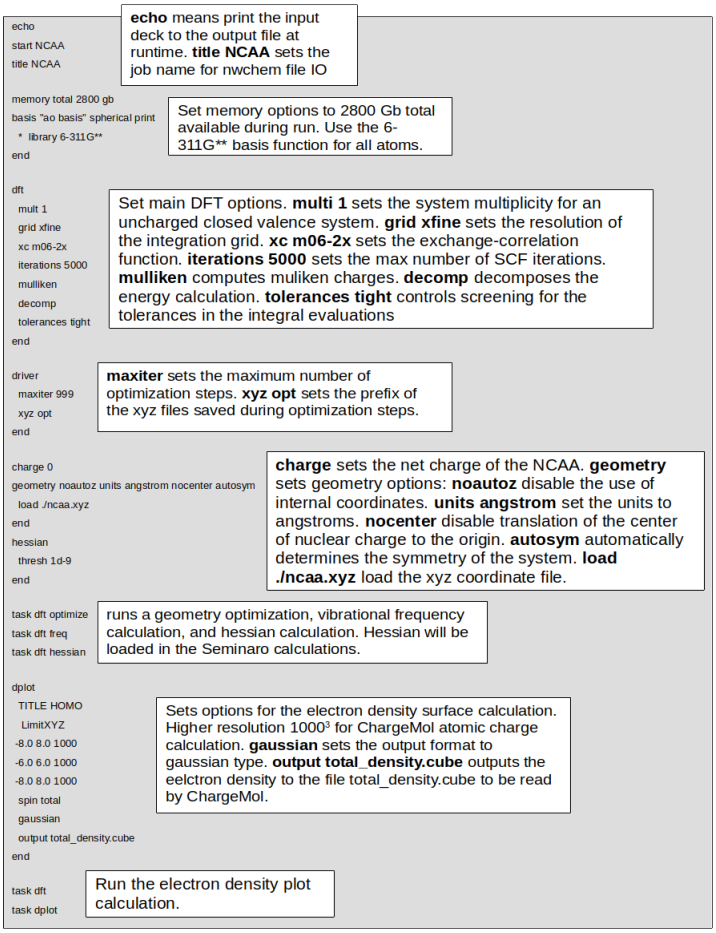

Figure 2: NWChem Input File

Opening the `ncaa.nw` file you will find several options and tasks listed to run a quantum chemical calculation outlined in figure 2. The overall goal of this calculation is shown in figure 1 where NWChem optimizes the geometry of the molecule, computes the vibrational spectrum, Hessian, and electron density surface. The geometry optimization describes the lowest energy conformation of the molecular topology bond lengths and angles. After the geometry optimization the vibrational frequencies are computed to verify the calculation converged (no imaginary frequencies). The hessian is output to compute force field parameters using the Seminario method. To start the NWChem calculation run the following.

```
$ nwchem ncaa.nw > ncaa.out &
```

Note: This calculation can take several hours depending on the size of the NCAA. The electron density calculation requires a lot of system memory by default to reliably calculate net atomic charges with ChargeMol.

To reduce memory requirements for this step decrease the 1000 Å^3 grid resolution under the `dplot` options. While this job runs you can monitor progress using the grep command:

```
$ grep \@ ncaa.out
@ Step       Energy          Delta E   Gmax     Grms     Xrms     Xmax     Walltime
@ ---- ---------------- -------- -------- -------- -------- -------- ---------
@    0   -957.55358468  0.0D+00  0.03916  0.01136  0.00000  0.00000      81.7
@    1   -957.56626592 -1.3D-02  0.01875  0.00476  0.04035  0.21940     185.7
@    2   -957.56912597 -2.9D-03  0.01002  0.00352  0.01755  0.10763     303.5
@    3   -957.56998882 -8.6D-04  0.01013  0.00306  0.01002  0.04192     419.9
@    4   -957.57154232 -1.6D-03  0.00547  0.00215  0.01549  0.06607     534.7
@    5   -957.57302184 -1.5D-03  0.00781  0.00236  0.02543  0.07702     631.5
@    6   -957.57425940 -1.2D-03  0.00606  0.00202  0.02986  0.09955     732.5
@    7   -957.57530360 -1.0D-03  0.00839  0.00220  0.02545  0.06554     827.7
@    8   -957.57606212 -7.6D-04  0.00690  0.00187  0.02205  0.06115     889.5
@    9   -957.57654234 -4.8D-04  0.00690  0.00179  0.01745  0.06265     950.6
@   10   -957.57706316 -5.2D-04  0.00442  0.00127  0.02384  0.08106    1046.2

$ grep 'Optimization converged' ncaa.out
      Optimization converged

#Checking for convergence. No negative or imaginary frequencies.
$ grep -A 10 'Projected Infra Red' ncaa.out
 Normal Eigenvalue ||           Projected Infra Red Intensities
  Mode    [cm**-1]  || [atomic units] [(debye/angs)**2] [(KM/mol)] [arbitrary]
 ------ ---------- || -------------- ----------------- ---------- -----------
    1       -0.000 ||     0.000230          0.005          0.225       0.083
    2       -0.000 ||     0.000136          0.003          0.133       0.049
    3       -0.000 ||     0.000246          0.006          0.239       0.089
    4       -0.000 ||     0.000042          0.001          0.041       0.015
    5        0.000 ||     0.000167          0.004          0.163       0.060
    6        0.000 ||     0.000077          0.002          0.075       0.028
    7       32.686 ||     0.000998          0.023          0.973       0.360
    8       58.183 ||     0.000236          0.005          0.230       0.085
```

<net charge>
0
</net charge>

Net charge of the NCAA

<periodicity along A, B, and C vectors>
.true.
.true.
.true.
</periodicity along A, B, and C vectors>

<charge type>
DDEC6
</charge type>

Type of net atomic charges to calculate.

<compute BOs>
.true.
</compute BOs>

<input filename>
total_density.cube
</input filename>

Electron density surface input file: total_density.cube

<atomic densities directory complete path>
../atomic_densities/
</atomic densities directory complete path>

Path to the location of the ChargeMol Atomic Density files.

Figure 3: ChargeMol Input File

Once this calculation is complete you should see the `NCAA.hess` and `total_density.cube` files in your working directory. Net atomic charges can now be computed using the ChargeMol program. The input file for this program is annotated in figure 3. Copy or create a symbolic link to the atomic densities distributed with the chargemol code to the `../atomic_densities/` folder.

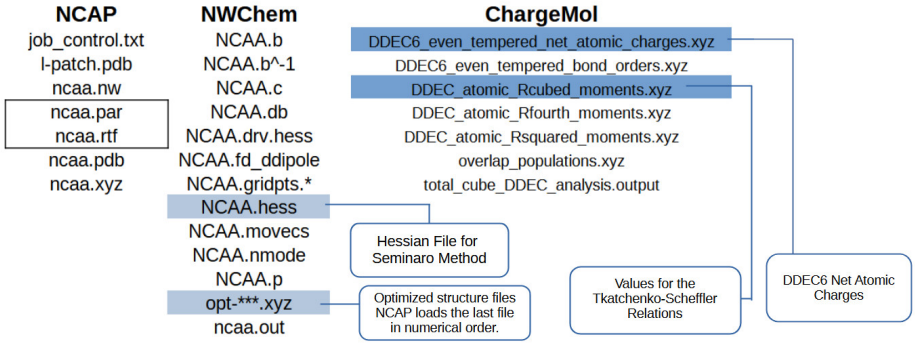

Figure 4: Output Files

After the ChargeMol calculation is complete you should see a complete set of output files annotated in figure 5. At this stage NCAP can be run again to compile parameter and topology files for molecular

dynamics calculations.

```
$ ncap 5ftrp
```

During the second NCAP execution the output files will be imported and force-field parameters computed. The molecular structure is automatically parsed for topology generation and tailored towards in-silico mutations in VMD. At the final stage of parameter generation the 2D Correction Map (CMAP) for protein backbone is computed using XTB. NCAP will automatically generate input structures and input files for XTB and compile the output of each of the calculations. The CMAP calulation will take approximately 45 minutes depending on your system configuration. Once NCAP finishes the calculation your `ncaa.rtf` and `ncaa.par` will be output. From here we can mutate a canonical peptide with these output files in VMD.

```
package require psfgen
package require autoionize
package require solvate

topology top_all27_prot_na.rtf
topology ncaa.rtf

segment A {
    pdb p12w.pdb
    mutate 6 NCA
}

coordpdb p12w.pdb A
guesscoord
writepdb tc5b_mut.pdb
writepsf tc5b_mut.psf
#solvate tc5b_mut.psf tc5b_mut.pdb -t 8 -o tc5b_mut_wb
#autoionize -psf tc5b_mut_wb.psf -pdb tc5b_mut_wb.pdb \
-sc 0.02 -cation POT -o tc5b_mut_wbi

exit
```

```
$ vmd -dispdev text -e tc5b_psfgen.tcl
```

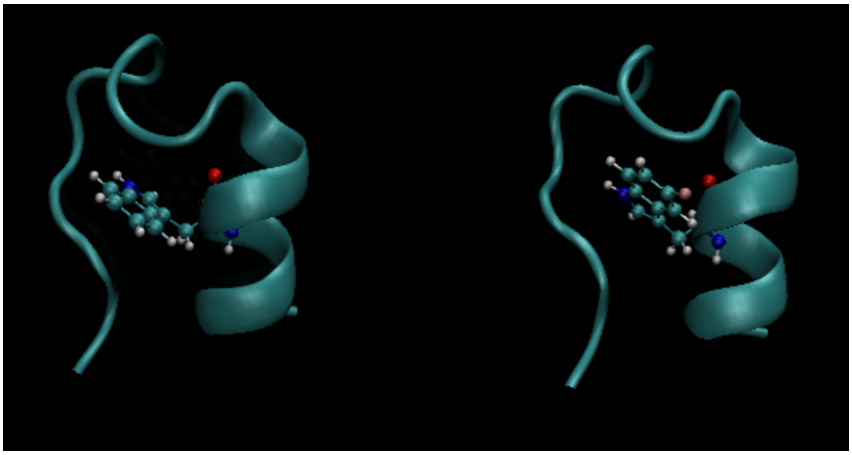

Figure 5: VMD in-silico mutation

```
structure          ./tc5b_mut.psf
coordinates        ./tc5b_mut.pdb

rigidBonds all

firsttimestep      0

paraTypeCharmm      on
parameters          ./par_all27_prot_na.prm
parameters          ./ncaa.par

exclude             scaled1-4
1-4scaling          1.0
cutoff              12.0
switching           on
switchdist          10.0
pairlistdist        14.0

temperature         300

langevin on
langevinTemp 300

outputName          pc
```

```
restartfreq          500
dcdfreq              250
xstFreq              250
outputEnergies       100

binaryrestart no
binaryoutput no

GBIS on

minimize 5000
run 1000000
```

With the force field parameters generated we can run a NAMD simulation of the mutation. The `p12w.conf` files is listed above to run a 1 ns simulation of the peptide in GBIS solvent. Note: this simulation is designed to be a quick demonstration. For better accuracy you will need to add TIP3 solvent and neutralize the system. See the commented commands in `tc5b_psfgen.tcl`.

```
$ namd2 p12w.conf > p12w.out
```