Quantum mechanical calculation of atomic polarizabilties

Written by Esther Heid, Institute of Computational Biological Chemistry, University of Vienna

This file explains how to calculate atomic polarizabilities using Gaussian09 or Psi4 based on the electron density or the electrostatic potential, as well as how to produce ab initio quality polarizabilities employing a neural net.

If you use one of the software packages or scripts developed in our group, please cite:

- **Atomic polarizabilities script: Ref. [1,2]**
- **ESP polarizabilities script: Unpublished results, please check again later**
- **Electrostatic parameter predictor (increment or neural net algorithm): Ref. [3]**

Please also make sure to cite any packages of other groups, such as Gaussian09, Psi4, GDMA, etc. For questions or feedback regarding this tutorial please contact:

Esther Heid (esther@mdy.univie.ac.at) , or, if unavailable,

Christian Schröder (christian@mdy.univie.ac.at)

Contents:

1 Atomic polarizabilities from electron densities

Requirements:

- Gaussian or Psi4, or any QM program that can produce `fchk` files
- GDMA (download executable from http://www-stone.ch.cam.ac.uk/pub/gdma/)
- Atomic polarizabilities python script (download from https://www.mdy.univie.ac.at/resources/polarizabilities/atomic_polarizabilities.html)
- Python3 (or python2 if you import the respective print-functions)

1.1 Mathematical background

For an in-detail description, please refer to Ref. [1,2,4] and references therein. In general, the polarizability tensor α can be computed as either the first derivative of the dipole moment μ or the second derivative of the energy E with respect to the electric field $\mathbf{F}$ as

$$\alpha_{ab} = -\frac{\partial^2 E}{\partial F_a \partial F_b}\bigg|_{F_a=F_b=0} = \frac{\partial \mu_a}{\partial F_b}\bigg|_{F_b=0} \tag{1}$$

with a and b denoting the x, y and z directions. Since we are interested in atomic polarizabilities, the molecular dipole moment μ needs to be dissected into atomic contributions μ_i of each non-overlapping atomic integration basin Ω_i of atom i. The changes of μ_i in response to an applied electric field give rise to atomic polarizabilities α_i. Although the molecular dipole moment $\mu = \sum_i q_i R_i$ is origin independent for neutral molecules, the atomic contributions $q_i R_i$ are not, since the individual partial charges q_i at the atomic coordinates R_i are usually not zero. Bader *et al.* introduced an origin-independent definition of the atomic contributions to the dipole moment for neutral molecules without rings or cages. [5, 6] This approach was generalized by Keith to account for rings and cages in neutral molecules, too, via the definition of bond contributions. [7]
The origin independent atomic dipole moment μ_i of an atomic site i at coordinates R_i bonded to N_b sites j is then

$$\boldsymbol{\mu}_i = \sum_{j=1}^{N_b} q_{b(ij)}(R_i - R_{b(ij)}) + \int_{\Omega_i} \rho(r) \cdot (r - R_i) dr = \mu_{ic} + \mu_{ip}. \tag{2}$$

where $q_{b(ij)}$ is the bond charge contribution of the directed bond between i and j to the net partial charge of i. $R_{b(ij)}$ denotes the coordinates of the bond charge (this can be set to $(R_i + R_j)/2$). The latter term in Eq. 2, μ_{ip}, describes the polarization of the electron cloud around a nucleus. The polarization contribution was obtained from the GDMA code of Misquitta and Stone, [8,9] where multipoles of order 1 (dipoles) were calculated for each atomic site. The analysis was carried out with the newer grid-based algorithm, which is the default in version 2.2.11, featuring a better convergence for diffuse basis sets. The former term, the contribution of charge transfer to the dipole moment μ_{ic}, is obtained from the charges and coordinates of each atomic site. The net atomic charges were taken from GDMA. The bond charges for a neutral molecule containing N atoms,

N_B bonds and N_R rings arise from the partial charges by simple linear algebra by solving the set of equations

$$q_i = \sum_{j=1}^{N_b} q_{b(ij)} \tag{3}$$

$$q_{b(ij)} = -q_{b(ji)} \tag{4}$$

$$\sum_{i,j=1+i}^{ring} q_{b(ij)} = 0. \tag{5}$$

Eq. (3) states that each partial charge q_i can be expressed as a sum of bond contributions $q_{b(ij)}$ (N equations). Since the bond charges are directed bonds, Eq. (4) must apply (N_B equations). In a ring, the summed up bond contributions in one direction must equal zero, Eq. (5) (N_R equations). Thus, $N + N_B + N_R$ linear equations need to be solved for $2N_B$ unknown $q_{b(ij)}$. However, since all charges q_i must add up to zero, one equation is redundant. Furthermore, if the molecule contains N_C cages, another equation per cage is redundant, too. Hence, one ends up with $2N_B$ linearly independent equations, because $N + N_B + N_R - 1 - N_C$ equals $2N_B$ by invocation of the Poincáre-Hopf relationship (see also Ref. 5). Thus, for any molecule of arbitrary structure, there exits exactly one unique solution of $q_{b(ij)}$. [7]

Although this procedure yields quantum mechanical atomic polarizabilities of neutral molecules, it cannot be used for charged molecules without modification, since bond charges for charged species are ill-defined. However, a set of different bond charges can be obtained by subtracting the net charge of the ion (q'_i=$q_i - q/N$) and thereby removing the origin dependence of the overall dipole moment. The new charges q'_i lead to a new set of bond charges $q_{b(ij)}$ We thus get

$$q'_i = \sum_{j=1}^{N_b} q'_{b(ij)} = q_i - \frac{q}{N} = \sum_{j=1}^{N_b} q_{b(ij)} - \frac{q}{N} \tag{6}$$

Since the atomic polarizabilities are derived from the derivative of the dipole moment with electric field, the property of interest is $\Delta\mu_{a,i} = \mu_{a,i}(F_b) - \mu_{a,i}(-F_b)$ which is derived from Δq_i and thus $\Delta q_{b(ij)}$. Since $\Delta q_i = \Delta q'_i$, the change in dipole moment is not affected by subtraction of the netto charge and $\Delta\mu_i = \Delta\mu'_i$. Thus, calculating μ'_{ic} using the reduced charges $q'_i = q_i - \frac{q}{N}$ produces the correct atomic polarizabilities $\alpha_{ab,i}$ of the charged molecule. Please note that the charge is only subtracted during the analysis routine, namely the calculation of μ_c and never actually subtracted in the QM calculation or GDMA analysis.

1.2 Case study 1a: G09 MO6-2X calculation of acetate

Polarizabilities should always be calculated on an optimized geometry. In the following, we work on acetate, which was optimized at the MP2/6-31+G(d) level of theory. Six single-point calculations of acetate with external electric fields in the positive and negative x, y and z direction need to be run. Create input files, here for example for an electric field in the positive x direction and name the file for example `x.inp`. Electric fields of 0.0008 au have proven to work well:

```
%Chk=x
#p <method>/<basis_set> field=x+8 density=current

Numerical calculation of atomic polarizabilities

-1 1
C 0.613146075611 1.296817574710 -0.001408652073
C -0.054498764942 -0.101727893662 -0.001088564526
H -0.135232499598 2.094735817787 -0.000936517573
H 1.255589748430 1.392531005997 -0.884205440087
H 1.256637717162 1.392473067087 0.880634402951
O -1.324128058114 -0.123476765848 0.001602292852
O 0.755237071679 -1.080593442882 0.000555231991
```

Choose a suitable method and basis set. The test calculation have been run using the DFT functional M06-2X and Sadlej's polarizable PVTZ basis set, which can be downloaded from the EMSL Basis Set Exchange website. Run the calculation via

```
$ g09 x.inp
```

and transform the file `x.chk` to `x.fchk` via

```
$ formchk x.chk x.fchk
```

In the same way, create calculations in the five other directions. You can copy `x.inp` and change the header, for example for the negative y direction create a file `my.inp` with the header

```
%Chk=my
#p <method>/<basis_set> field=y-8 density=current
```

Continue until all six `fchk` files are created. Now, we can feed the `fchk` files to GDMA. Create the directory `gdma/out`

```
$ mkdir -p gdma/out
$ cd gdma/out
```

and create six GDMA input files for each external field. For example, the file `x.inp` tells GDMA to use the `SCF density` from the file `../../x.fchk`.

```
File ../../x.fchk DENSITY SCF

Angstrom
Multipoles
  Limit 1
Start

Finish
```

Run via

```
$ <PATH TO GDMA EXECUTABLE> < x.inp > x.out
```

Repeat for the five other directions. The output in `x.out` should read

```
C x = -1.346377 y = -0.060108 z = 0.000151 angstrom
         Maximum rank = 1 Radius = 0.650 angstrom
                Q00 = -0.124538
```

```
|Q1| = 0.144189 Q10 = 0.000066 Q11c = 0.143297 Q11s = 0.016016

C x = 0.202110 y = 0.002068 z = -0.001243 angstrom
          Maximum rank = 1 Radius = 0.650 angstrom
                Q00 = 0.187224
|Q1| = 0.122695 Q10 = 0.000461 Q11c = 0.122693 Q11s = 0.000606

H x = -1.702832 y = -1.094362 z = 0.000614 angstrom
          Maximum rank = 1 Radius = 0.325 angstrom
                Q00 = 0.050291
|Q1| = 0.183317 Q10 = -0.000054 Q11c = 0.133371 Q11s = 0.125766

H x = -1.731829 y = 0.463772 z = -0.882006 angstrom
          Maximum rank = 1 Radius = 0.325 angstrom
                Q00 = 0.051458
|Q1| = 0.182324 Q10 = 0.117055 Q11c = 0.127014 Q11s = -0.058376

H x = -1.730286 y = 0.464027 z = 0.882833 angstrom
          Maximum rank = 1 Radius = 0.325 angstrom
                Q00 = 0.051643
|Q1| = 0.182194 Q10 = -0.117126 Q11c = 0.126773 Q11s = -0.058349

O x = 0.813848 y = -1.110682 z = 0.000321 angstrom
          Maximum rank = 1 Radius = 0.650 angstrom
                Q00 = -0.605842
|Q1| = 0.571151 Q10 = -0.000215 Q11c = -0.213034 Q11s = 0.529934

O x = 0.689970 y = 1.175032 z = 0.000318 angstrom
          Maximum rank = 1 Radius = 0.650 angstrom
                Q00 = -0.610235
|Q1| = 0.571924 Q10 = -0.000258 Q11c = -0.162825 Q11s = -0.548257

Total multipoles referred to origin at
          x = 0.000000, y = 0.000000, z = 0.000000 angstrom
                Q00 = -0.999999
|Q1| = 1.562646 Q10 = -0.000831 Q11c = -1.560853 Q11s = -0.074827
```

GDMA outputs the coordinates of each atom, the polarization term of the atomic dipole moment, as well as the charge around each atom. Now, leave the `out` directory

```
$ cd ../
```

and create a file `connec.inp` which lists the connectivities of a molecule in the syntax `bond <atom1> <atom2>` for a bond between atom1 and atom2, as well as `ring <atoms>`, e.g. `ring <atom1> <atom2> <atom3> <atom4>` for a ring between atom1, atom2, atom3 and atom4 (where atom1 and atom4 are also connected). The order in which the bonds or atoms are given does not matter. For acetate, this reads for example

```
bond 1 3
bond 2 6
bond 1 4
bond 2 7
bond 1 5
bond 1 2
```

You can now run the analysis script via

```
$ python atomic_polarizabilities_charge.py
```

which will ask for the file locations, field strengths (enter ’0.0008’) and whether the output should be changed to cubic Angstrom (enter ’yes’). The relevant part of the output should read

```
Total polarizability:
Name a
   C 1.12
   C 1.25
   H 0.44
   H 0.45
   H 0.45
   O 1.41
   O 1.41
Summed up contributions:
6.52457921911
```

which are the atomic polarizabilities of the atoms in acetate. Congratulations, you just finished your first polarizability calculation. Let us now try to use a different method.

1.3 Case study 1b: G09 MP2 calculation of acetate

Redo Case Study 1a, but employing MP2 instead of DFT. Change the headers of the Gaussian input files. In the GDMA input, change ’DENSITY SCF’ to ’DENSITY MP2’. Redo all calculations. The analysis should yield

```
Total polarizability:
Name a
   C 1.13
   C 1.34
   H 0.45
   H 0.46
   H 0.46
   O 1.54
   O 1.54
Summed up contributions:
6.91528381492
```

1.4 Case study 1c: Psi4 MP2 calculation of acetate

Instead of Gaussian, we can also use different QM packages to create the `fchk` files. For example, using Psi4, all six calculations can be done using a single input file as

```
molecule {
-1 1

    C 0.613146075611 1.296817574710 -0.001408652073
    C -0.054498764942 -0.101727893662 -0.001088564526
    H -0.135232499598 2.094735817787 -0.000936517573
    H 1.255589748430 1.392531005997 -0.884205440087
    H 1.256637717162 1.392473067087 0.880634402951
    O -1.324128058114 -0.123476765848 0.001602292852
    O 0.755237071679 -1.080593442882 0.000555231991

symmetry c1
}

set {
    basis Sadlej
    scf_type df
    df_basis_scf def2-tzvpp-jkfit
    df_basis_mp2 def2-tzvppd-ri
}

set perturb_h true
set perturb_with dipole
set perturb_dipole [0.0008, 0, 0]
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'x.fchk')

set perturb_h true
set perturb_with dipole
set perturb_dipole [-0.0008, 0, 0]
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'mx.fchk')

set perturb_h true
set perturb_with dipole
set perturb_dipole [0, 0.0008, 0]
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'y.fchk')

set perturb_h true
set perturb_with dipole
set perturb_dipole [0, -0.0008, 0]
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'my.fchk')

set perturb_h true
set perturb_with dipole
set perturb_dipole [0, 0, 0.0008]
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'z.fchk')

set perturb_h true
set perturb_with dipole
set perturb_dipole [0, 0, -0.0008]
```

```
grad, wfn = gradient('mp2', return_wfn=True)
fchk(wfn,'mz.fchk')
```

where we chose RI-MP2 with Sadlejs PVTZ basis set. This is not a default basis set, so make sure you have downloaded the basis set from EMSL basis set exchange and saved it to the respective Psi4 installation folder (see Psi4 documentation) or to the current directory. Name the basis set file `sadlej.gbs`. Then, run the Psi4 calculation via

```
$ psi4 -i psi4.inp -o psi4.log
```

where we have named the input file `psi4.inp`. This should produce all six `fchk` files. Run GDMA (change the density to 'DF-MP2') and the python script, to obtain

```
 Total polarizability:
Name a
   C 1.13
   C 1.34
   H 0.45
   H 0.46
   H 0.46
   O 1.54
   O 1.54
Summed up contributions:
6.91046592987
```

1.5 Case study 1d: Sum hydrogen polarizabilities into nonhydrogen atoms

If the polarizabilities of the hydrogen atoms should be summed into the respective nonhydrogen atoms, this can be simply done by summing up the polarizabilities after the analysis. If you still want the analysis itself to exclude H atoms, the GDMA input needs to be modified. Inside Case Study 1c, create an additional directory

```
$ mkdir -p gdma_no_h/out
$ cd gdma_no_h/out
```

and add the line 'Delete H' to each GDMA input file:

```
File ../../x.fchk DENSITY DF-MP2

Angstrom
Multipoles
  Delete H
  Limit 1
Start

Finish
```

Since the H have been deleted from the molecule, the file `connec.inp` needs to change, where also the atom numbers have changed:

```
bond 1 2
bond 2 3
bond 2 4
```

Rerun the python script to obtain

```
Total polarizability:
Name a
  C 2.45
  C 1.36
  O 1.55
  O 1.55
Summed up contributions:
6.91046779498
```

Congratulations, you can now conduct QM calculations in Gaussian and Psi4 and calculate the numerical atomic polarizabilities based on the electron density at different external fields. Before we move on to the next chapter, let us quickly summarize the pros and cons for this method.

1.6 Pros and Cons

Pros:

- Summing up the atomic polarizabilities must recover the QM molecular polarizabilities
- All atoms (both buried and exposed atoms) in a molecule are equally important during the analysis routine

Cons:

- Method depends on the basis set (although weakly, assumed that the basis set is large enough to reproduce electron movement in response to an electric field).
- QM polarizabilities are usually too large to be used in molecular dynamics simulation, and need to be scaled. Refer to Ref. [10] for scaling factors.
- Computationally expensive for very large molecules

2 Atomic polarizabilities from the electrostatic potential

Requirements:

- Psi4 and RESP module to produce a grid (https://github.com/cdsgroup/resp)
- Python script to evaluate atomic polarizabilities (not published yet. Use `pol_esp.py` at your own risk. This script is still under development!)
- Python3 (or python2 if you import the respective print-functions)

2.1 Mathematical background

Atomic polarizabilities can be evaluated via the induced electrostatic potential $\Delta\Phi$ at gridpoints n when an external electric field $\mathbf{F}$ is applied as

$$\Delta\Phi_n = \Phi_n(\mathbf{F}) - \Phi_n(\mathbf{0}). \tag{7}$$

Each $\Delta\Phi_n$ can be represented by induced dipole moments $\Delta\boldsymbol{\mu}_i(\mathbf{F})$ at polarizable centers i (here the nuclei) as

$$\Delta\Phi_n = \frac{1}{4\pi\epsilon_0}\frac{\Delta\boldsymbol{\mu}_i(\mathbf{F})\cdot\mathbf{r}_{i,n}}{r_{i,n}^3} \tag{8}$$

where $\mathbf{r}_{i,n}$ is the distance vector between atom i and gridpoint n, and $r_{i,n}$ its length. ϵ_0 is the vacuum permittivity. The elements of the atomic polarizability tensors can then be obtained via

$$\alpha_{i,ab} = \frac{\Delta\boldsymbol{\mu}_{i,a}(F_b)}{F_b} \tag{9}$$

where a and b denote the x, y and z direction.

2.2 Case study 2a: Psi4 MP2 calculation of acetate

Create a new directory. First, a set of gridpoints needs to be obtained. We use 20 equally spaced gridpoint per Å^2 on four surfaces around the molecule at 1.4, 1.6, 1.8 and 2.0 times the van der Waals radii of the atoms. Currently, we use the RESP utility to produce the grid, but an own function to produce the grid is currently under development. For now, create a directory `resp`

```
$ mkdir resp
$ cd resp
```

and set up an input file, such as

```
from __future__ import division, absolute_import, print_function

def calculate():
    import psi4
    import resp

    mol = psi4.geometry(
   """
-1 1
    C 0.613146075611 1.296817574710 -0.001408652073
    C -0.054498764942 -0.101727893662 -0.001088564526
    H -0.135232499598 2.094735817787 -0.000936517573
    H 1.255589748430 1.392531005997 -0.884205440087
    H 1.256637717162 1.392473067087 0.880634402951
    O -1.324128058114 -0.123476765848 0.001602292852
    O 0.755237071679 -1.080593442882 0.000555231991
"""
    )

    mol.update_geometry()

    options = {'N_VDW_LAYERS' : 4,
              'VDW_SCALE_FACTOR' : 1.4,
              'VDW_INCREMENT' : 0.2,
              'VDW_POINT_DENSITY' : 20.0,
              'resp_a' : 0.0005,
              'RESP_B' : 0.1,
              'BASIS_ESP':'3-21G',
              'METHOD_ESP':'HF',
              'RADIUS':{'BR':1.97,'I':2.19}
    }

    # Call for first stage fit
    charges1 = resp.resp([mol], [options])


calculate()
```

and run the script in python. You can choose any basis set or method, as well as any resp parameters (`resp_a` and `resp_b`) since we are solely interested in the produced grip (and not the RESP charges). This produces a file `1_default_grid.dat`. Leave the directory and copy the grid:

```
$ cd ../
$ cp resp/1_default_grid.dat grid.dat
```

Now, produce Psi4 input files for electric fields in the positive x, y and z directions, as well as without a field, here for example for the x direction (file named `x.inp`):

```
molecule {
-1 1
   C 0.613146075611 1.296817574710 -0.001408652073
   C -0.054498764942 -0.101727893662 -0.001088564526
   H -0.135232499598 2.094735817787 -0.000936517573
   H 1.255589748430 1.392531005997 -0.884205440087
   H 1.256637717162 1.392473067087 0.880634402951
   O -1.324128058114 -0.123476765848 0.001602292852
   O 0.755237071679 -1.080593442882 0.000555231991
symmetry c1
}
```

```
set {
   basis Sadlej
   e_convergence 6
   d_convergence 8
   scf_type df
   df_basis_scf def2-tzvpp-jkfit
   df_basis_mp2 def2-tzvppd-ri
}

set perturb_h true
set perturb_with dipole
set perturb_dipole [0.0008, 0, 0]
property('mp2', properties=['grid_esp','dipole'])
```

Run with Psi4 as

```
$ psi4 -i x.inp -o x.log
```

and rename the calculated ESP

```
$ mv grid_esp.dat x.esp
```

In the same way produce `y.esp`, `z.esp` and `0.esp`. Furthermore, you need a file `coor.xyz` where you copy the atomic coordinates as

```
   C 0.613146075611 1.296817574710 -0.001408652073
   C -0.054498764942 -0.101727893662 -0.001088564526
   H -0.135232499598 2.094735817787 -0.000936517573
   H 1.255589748430 1.392531005997 -0.884205440087
   H 1.256637717162 1.392473067087 0.880634402951
   O -1.324128058114 -0.123476765848 0.001602292852
   O 0.755237071679 -1.080593442882 0.000555231991
```

Now, run the `pol_esp.py` script, which reads the four ESP files, `grid.dat` and `coor.xyz` and calculates the atomic polarizabilities at each non-hydrogen atom *i*. You should obtain

```
 Name a_xx a_yy a_zz a_iso
-----------------------------------------
    C 2.433 3.246 2.606 2.762
    C 1.605 0.716 -0.557 0.588
    O 1.936 1.187 1.293 1.472
    O 1.273 1.986 1.459 1.573
```

2.3 Pros and Cons

Pros:

- Method depends to a less extent on the basis set than the electron density routine
- Smaller polarizabilities than usual QM polarizabilities are obtained, so that scaling for use in molecular dynamics simulation may become obsolete.

Cons:

- Summing up the atomic polarizabilities does not recover the QM molecular polarizabilities; usually a smaller value is obtained
- Mainly exposed atoms contribute to a change in electrostatic potential, thus buried atoms are usually not represented well
- Computationally expensive for very large molecules
- Many sets of induced dipole moments may exist, which produce the same induced potential. There is no guarantee that the least-squares linear solver finds the best solution / converges.

3 Atomic polarizabilities via a neural net prediction

Requirements:

- Predictor module, download from
  www.mdy.univie.ac.at/resources/electrostatic_parameter/prediction.html
- Python3

3.1 Mathematical background

See Ref. [3] for a detailed description. Each atom is assigned an atom type. Based on atom connectivities, an atomic structure vector is set up for each atom, where the identity of the atom and its surroundings is described. Based on these structure vectors atomic polarizabilities (and charges) are predicted.

3.2 Case study 3a: Predicting atomic polarizabilities of acetate

Set up a mol2 file of acetate

```
@<TRIPOS>MOLECULE
 Molden generated mol2
    7 6 1
 SMALL
 NO_CHARGES
 ****
 ****
@<TRIPOS>ATOM
    1 C 0.6131 1.2968 -0.0014 C.3 1 RES1 0.0000
    2 C -0.0545 -0.1017 -0.0011 C.2 1 RES1 0.0000
    3 H -0.1352 2.0947 -0.0009 H 1 RES1 0.0000
    4 H 1.2556 1.3925 -0.8842 H 1 RES1 0.0000
    5 H 1.2566 1.3925 0.8806 H 1 RES1 0.0000
    6 O -1.3241 -0.1235 0.0016 O.co2 1 RES1 0.0000
    7 O 0.7552 -1.0806 0.0006 O.co2 1 RES1 0.0000
@<TRIPOS>BOND
    1 1 2 1
    2 1 3 1
    3 1 4 1
    4 1 5 1
    5 2 6 1
    6 2 7 2
@<TRIPOS>SUBSTRUCTURE
     1 RES1 1
```

Upload on https://cgenff.umaryland.edu/ to produce a str file, save to acet.str. The file should look like

```
 * Toppar stream file generated by
* CHARMM General Force Field (CGenFF) program version 2.2.0
* For use with CGenFF version 4.0
```

```
*

read rtf card append
* Topologies generated by
* CHARMM General Force Field (CGenFF) program version 2.2.0
*
36 1


RESI Molden -1.000 ! param penalty= 0.000 ; charge penalty= 0.000
GROUP ! CHARGE CH_PENALTY
ATOM C1 CG331 -0.370 ! 0.000
ATOM C2 CG2O3 0.620 ! 0.000
ATOM H1 HGA3 0.090 ! 0.000
ATOM H2 HGA3 0.090 ! 0.000
ATOM H3 HGA3 0.090 ! 0.000
ATOM O1 OG2D2 -0.760 ! 0.000
ATOM O2 OG2D2 -0.760 ! 0.000

BOND C1 C2
BOND C1 H1
BOND C1 H2
BOND C1 H3
BOND C2 O1
BOND C2 O2
IMPR C2 O2 O1 C1

END

read param card flex append
* Parameters generated by analogy by
* CHARMM General Force Field (CGenFF) program version 2.2.0
*

BONDS

ANGLES

DIHEDRALS

IMPROPERS

END
RETURN
```

Now, run the predictor script via

```
 ./<path to predictor directory>/predictor.sh mol2 acet.mol2
```

which produces the following polarizabilities using the increment scheme

```
         C1 CG331 1.077
         C2 CG2O3 1.315
```

```
        H1 HGA3 0.390
        H2 HGA3 0.390
        H3 HGA3 0.390
        O1 OG2D2 1.380
        O2 OG2D2 1.380
```

and using the neural net

```
        C1 CG331 1.126
        C2 CG2O3 1.337
        H1 HGA3 0.444
        H2 HGA3 0.444
        H3 HGA3 0.444
        O1 OG2D2 1.526
        O2 OG2D2 1.526
```

which are pretty close to the QM calculations from section 1 and 2.

3.3 Pros and Cons

Pros:

- Very fast, also for large molecules, since no QM calculations necessary
- No optimized geometry needs to be known, only atomic connectivities

Cons:

- Works only if CGenFF atom types can be assigned
- May not be accurate for very uncommon structure elements

4 References

[1] E. Heid, P. Hunt, and C. Schröder. "Evaluating excited state atomic polarizabilities of chromophores." *Phys. Chem. Chem. Phys.*, **20**(2018), 8554.

[2] E. Heid, A. Szabadi, and C. Schröder. "Quantum mechanical determination of atomic polarizabilities of ionic liquids." *Phys. Chem. Chem. Phys.*, **20**(2018), 10992.

[3] E. Heid, M. Fleck, P. Chatterjee, C. Schröder, and A. D. MacKerell, Jr. "Toward prediction of electrostatic parameters for force fields that explicitly treat electronic polarization." *J. Chem. Theory Comput.*, **15**(2019), 2460.

[4] E. Heid, M. Heindl, P. Dienstl, and C. Schröder. "Additive polarizabilities of halides in ionic liquids and organic solvents." *J. Chem. Phys.*, **149**(2018), 044302.

[5] R. F. W. Bader. *Atoms in Molecules: a Quantum Theory*. Oxford University Press, Oxford UK (1990).

[6] K. E. Laidig and R. F. W. Bader. "Properties of atoms in molecules: Atomic polarizabilities." *J. Chem. Phys.*, **93**(1990), 7213.

[7] T. A. Keith. *Atomic Response Properties*, pages 61–94. Wiley-VCH Verlag GmbH & Co. KGaA (2007).

[8] A. J. Stone. "Distributed multipole analysis: stability of large basis sets." *J. Chem. Theory Comput.*, **1**(2005), 1128.

[9] A. J. Misquitta and A. J. Stone. "Distributed polarizabilities obtained using a constrained density-fitting algorithm." *J. Chem. Phys.*, **124**(2006), 024111.

[10] J. A. Lemkul, J. Huang, B. Roux, and A. D. MacKerell Jr. "An empirical polarizable force field based on the classical drude oscillator model: Development history and recent applications." *Chem. Rev.*, **116**(2016), 4983.