MOMAP

Tutorial 08

Mobility Calculation

Version 2020A

June, 2021

MOMAP Tutorial 08

Version 2020A edited by:

Dr. Qikai Li

Released by Hongzhiwei Technology (Shanghai) Co., Ltd

and Z.G. Shuai Group

The information in this document applies to version 2020A of MOMAP

TABLE OF CONTENTS

MOMAP Mobility Calculation Tutorial

BACKGROUND

Due to the weak coupling between molecules, the charge transport in most organics is dominated by the hopping mechanism[1], which implies that the transport dynamics can be decomposed into elementary charge transfer processes between different pairs of molecules. The widely used charge transfer rate from the classical Marcus theory reads[2]

$$k_{ij} = \frac{V_{ij}^2}{\hbar}\sqrt{\frac{\pi}{\lambda k_B T}}\exp\left[-\frac{\left(\lambda + \Delta G_{ij}^0\right)^2}{4\lambda k_B T}\right]$$

Here, V_{ij} is the transfer integral between the initial and final states, λ is the reorganization energy which is defined as the energy change associated with the geometry relaxation during the charge transfer, and ΔG_{ij}^0 is relevant change of total Gibbs free energy. In molecular semiconductors with only one kind of molecules, ΔG_{ij}^0 equals to zero, and then the Marcus rate becomes

$$k_{ij} = \frac{V_{ij}^2}{\hbar}\sqrt{\frac{\pi}{\lambda k_B T}}\exp\left(-\frac{\lambda}{4k_B T}\right)$$

In 1993, based on the hopping model and Marcus theory, Shuai etc. proposed a quantum version of charge-transfer theory coupled with random-walk simulation of charge diffusion, it can well describe the "band-like" behavior under the hopping model. The quantum charge transfer rate is expressed as[3]

$$k_{if} = \frac{2|V_{fi}|^2}{\hbar^2}\int_0^\infty dt\,\exp\left[-\sum_j S_j\left(2n_j+1\right)\left(1-\cos\left(\omega_j t\right)\right)\right]\cos\left(\omega_{fi}t + \sum_j S_j sin(\omega_j t)\right)$$

For a particle in a known fixed position at $t = 0$, the central limit theorem tells us that after a large number of independent steps in the random walk, the walker's position is distributed according to a normal distribution of total *variance*:

$$\sigma^2 = \frac{t}{\delta t}\varepsilon^2,$$

where t is the time elapsed since the start of the random walk, ε is the size of a step of the random walk, and δt is the time elapsed between two successive steps. In three dimensions, the *variance* corresponding to the Green's function of the diffusion equation is:

$$\sigma^2 = 6\mathrm{D}t\,.$$

By equalizing this quantity with the variance associated to the position of the random walker, one obtains the equivalent diffusion coefficient to be considered for the asymptotic Wiener process toward which the random walk converges after a large number of steps:

$$\mathrm{D} = \frac{\sigma^2}{2n\mathrm{t}} = \frac{\varepsilon^2}{2n\delta t},$$

where n = 1, 2, or 3 is the dimensionality of the system under investigation.

The mobility of a charge carrier is related to the diffusion coefficient by the *Einstein* relationship:

$$\mathrm{D} = \frac{\mathrm{k}_B\mathrm{T}}{q}\mu\,.$$

Rearrange the above equation, we have:

$$\mu = \frac{q}{\mathrm{k}_B\mathrm{T}}\mathrm{D}\,.$$

Thus, the task is to first find the diffusion coefficient of a charge carrier, then the mobility of charge carrier is obtained by using the *Einstein* relationship. To fulfil the task, we can resort to Monte Carlo simulations.

The diffusion coefficient D is calculated through

$$\mathrm{D} = \frac{1}{2n}\frac{dMSD}{dt}\,.$$

where n = 1, 2, or 3 is the dimensionality of the system under investigation.

In MOMAP Transport package, we setup a lot of initial different (random) seeds, record the tracks, and then average over the tracks. Even though an individual track looks rather jittery, however, when a certain number of (say 2,000) tracks are averaged, we can get a pretty linear averaged line. With the averaged track, we can fit to a linear line, obtain the diffusion coefficient, and then the charge carrier mobility.

Reference:

1. L. J. Wang, G. J. Nan, X. D. Yang, Q. Peng, Q. K. Li, and Z. Shuai**, Chem. Soc. Rev. 39**, 423-434 (2010).
2. R. A. Marcus, **Rev. Mod. Phys. 65**, 599-610 (1993).
3. Guangjun Nan, Xiaodi Yang, Linjun Wang, Zhigang Shuai*, Yi Zhao, **Phys. Rev. B 79**, 115203~1-9 (2009).

QUICK START

To begin the transport mobility calculation with MOMAP, one needs two files, that is,

- Crystal structure file: either in `cif` or in `mol`
- MOMAP control file: `momap.inp`

Let's take naphthalene as an example, and suppose we have the `naphthalene.cif` file.

INFORMATION UNDER DATA DIRECTORY

First, we create a directory, called `naphthalene`, and copy the `naphthalene.cif` file to that directory, the directory may look like the following:

```
[momap]$ tree naphthalene
naphthalene
└── naphthalene.cif

0 directories, 1 file
[momap]$
```

The next step one needs to do is to create the MOMAP control file `momap.inp`, we have prepared a tool to generate the file, it is called `transport_geninp.exe`. With all the MOMAP transport programs, one can use `-h` or `--help` to see how to use the program as is the case for common Linux commands.

```
[momap]$ transport_geninp.exe -h
*********************************************************************************
*  MOMAP Transport Calculation Utility, Version 2021A (2.3.1) build on Mar 27 2021  *
*  Zhigang Shuai Group, Department of Chemistry, Tsinghua University, Beijing       *
*********************************************************************************
Transport Momap Input Generation

Usage: transport_geninp.exe [opts]
  -config s  : set config file, default to momap.inp
  -cif       : use cif file as molecule input (default)
  -mol       : use mol file as molecule input
  -module    : set to use environment module flag
  -terse     : generate terse momap.inp (default)
  -verbose   : generate verbose momap.inp
  -oldconfig : generate old config.inp
  -orca      : use ORCA compute engine
  -qchem     : use Q-Chem compute engine

e.g.: transport_geninp.exe
  transport_geninp.exe -config momap.inp
  transport_geninp.exe -verbose
  transport_geninp.exe -cif
  transport_geninp.exe -mol
  transport_geninp.exe -module
  transport_geninp.exe -config momap.inp -verbose
  transport_geninp.exe -oldconfig
  transport_geninp.exe -orca

[momap]$
```

As we are going to use `.cif` file as crystal file, and use Gaussian `g09/g16` as the QC compute engine, which is the default setting, we can run `transport_geninp.exe` in the current directory without any extra option, that is:

$ `transport_geninp.exe`

```
[naphthalene]$ ls
momap.inp       naphthalene.cif run.sh
[naphthalene]$
```

It generates two files: `momap.inp`, and `run.sh`, shown as follows:

```
[naphthalene]$ cat momap.inp
&transport
  do_transport_prepare              = 1
  do_transport_submit_HL_job        = 1
  do_transport_get_transferintegral = 1
  do_transport_submit_RE_job        = 1
  do_transport_get_re_evc           = 1
  do_transport_run_MC               = 1
  do_transport_get_mob_MC           = 1
  do_transport_run_MC_temp          = 0
  do_transport_get_mob_MC_temp      = 0
  do_transport_run_ME               = 0
  do_transport_get_mob_ME           = 0
  do_transport_run_ME_temp          = 0
  do_transport_get_mob_ME_temp      = 0
  do_transport_gather_momap_data    = 1

  # Job Scheduling
  sched_type      = pbs    ! pbs, slurm, lsf, or local
  queue_name      = workq

  compute_engine  = 1      ! 1 = Gaussian, 2 = ORCA, 3 = QCHEM
  qc_exe          = g09    ! g09/g16 or fullpath/orca or qchem

#  module_qc = gaussian/g09.e01
  qc_method       = b3lyp
  qc_basis        = b3lyp STO-3g
  qc_basis_re     = b3lyp STO-3g
  qc_memory       = 4096  ! MB
  qc_nodes        = 1
  qc_ppn          = 8

  temp            = 300

  # Temperature Dependence
  start_temp      = 200
  end_temp        = 300
  delta_temp      = 50

  ratetype        = quantum  ! marcus or quantum

  lat_cutoff      = 4       ! for neighbor list construction

  nsimu           = 2000
  tsimu           = 1000    ! in ns
  tsnap           = 5

  crystal         = naphthalene.cif
/
[naphthalene]$
```

```
[naphthalene]$ cat run.sh
#!/bin/sh

python $MOMAP_ROOT/bin/momap.py 1> log 2>&1 &

[naphthalene]$
```

One may need to modify the `momap.inp` according to one's specific situation. If used in a computing cluster environment, the module program is normally available, one may uncomment the `module_qc` line, that is, delete the # character, and change the line accordingly.

When the modification is done, one can run MOMAP transport package with the above script `run.sh`.

```
$ sh run.sh
```

Or

```
$ chmod +x run.sh
$ ./run.sh
```

If everything is fine, the calculation can be finished in less than 10 minutes on a 32-core computing node.

Now let's direct our attention to the `data` and `MC-quantum` directories as the key output data are put in there.

```
[naphthalene]$ ls data
NM-e.dat                 mol1_neighbors.cif       mol2_oniom.xyz           trans_int_files.dat
NM-h.dat                 mol1_neighbors_mid.cif   mol_lambda_4P-e.dat      uc_H.inp
VH.dat                   mol1_oniom.com           mol_lambda_4P-h.dat      uc_L.inp
VL.dat                   mol1_oniom.xyz           mol_lambda_evc-e.dat     unique_id_map.dat
cellvec.dat              mol2.mol                 mol_lambda_evc-h.dat     unitcell.cif
config.inp               mol2_neighbors.cif       neighbor.dat
mol1.mol                 mol2_neighbors_mid.cif   re_lambda_4P_files-e.dat
mol1_bonds.dat           mol2_oniom.com           re_lambda_4P_files-h.dat
[naphthalene]$
```

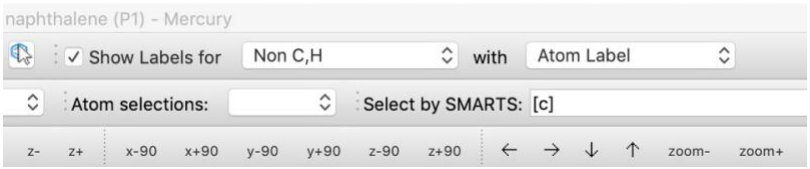
The running parameters actually used in the MOMAP transport calculation are put in file `data/config.inp`.

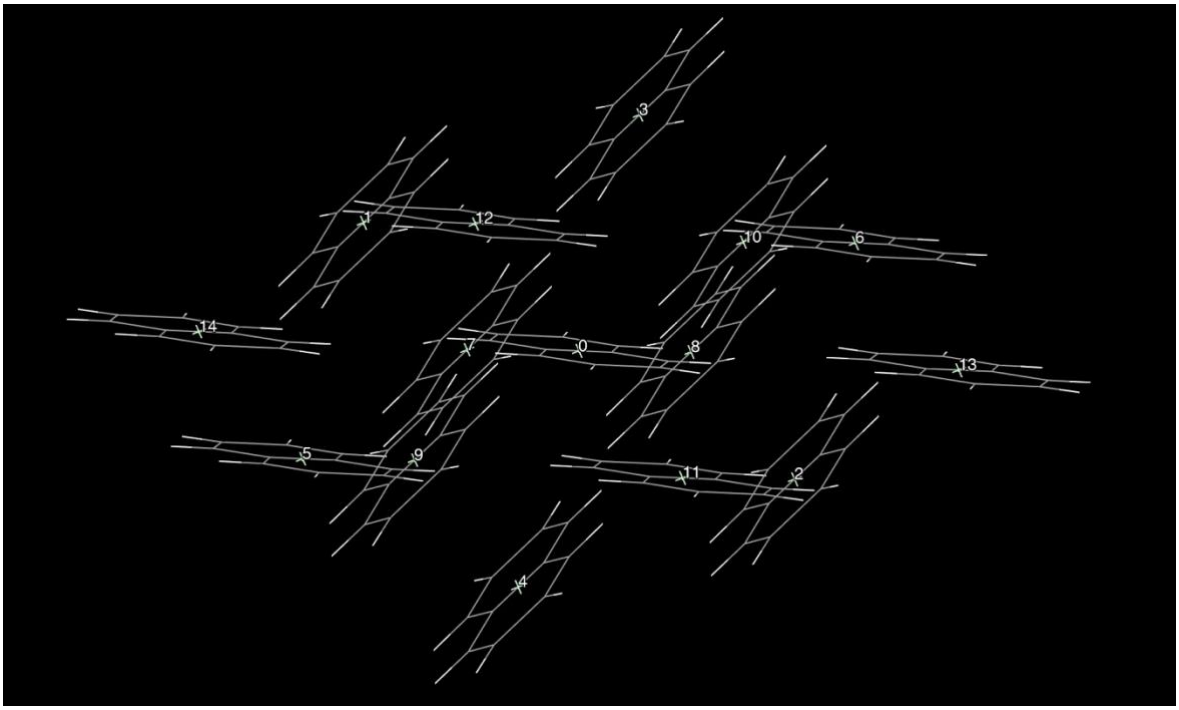
The `mol[1,2,…]_*` are the separated molecular information in the crystal structure file. Molecular neighbor information are in files `neighbor.dat` and `unique_id_map.dat`. One can also use `mol*_neighbors.cif` and `mol*_neighbors_mid.cif` to plot the central mol. plus the neighbor mol. figures, the latter `cif` files are with `mol_id` as virtual atoms. The `neighbor.dat` is shown as follows:

```
[data]$ cat neighbor.dat
# nroot, root_mol_info, nnei, neighbors_info
# r = molecular COM distance, cindex = neighbor cell index
2
1  -0.0000   0.0000   0.0000  # root
14 #      dx        dy        dz         r     cindex
   2   -4.0490   -2.9765   -0.0000    5.0253  -1 -1  0
   2    4.0490    2.9765   -0.0000    5.0253   0  0  0
   2   -4.0490    2.9765   -0.0000    5.0253  -1  0  0
   2    4.0490   -2.9765   -0.0000    5.0253   0 -1  0
   1    0.0000   -5.9530    0.0000    5.9530   0 -1  0
   1   -0.0000    5.9530    0.0000    5.9530   0  1  0
   2    0.8391   -2.9765   -7.1389    7.7799  -1 -1 -1
   2   -0.8391    2.9765    7.1389    7.7799   0  0  1
   2   -0.8391   -2.9765    7.1389    7.7799   0 -1  1
   2    0.8391    2.9765   -7.1389    7.7799  -1  0 -1
   1    4.8881    0.0000   -7.1389    8.6520   0  0 -1
   1   -4.8881   -0.0000    7.1389    8.6520   0  0  1
   1    4.8881    5.9530   -7.1389   10.5022   0  1 -1
   1   -4.8881   -5.9530    7.1389   10.5022   0 -1  1
2   4.0490   2.9765  -0.0000  # root
14 #      dx        dy        dz         r     cindex
   1    4.0490    2.9765    0.0000    5.0253   1  1  0
   1   -4.0490   -2.9765    0.0000    5.0253   0  0  0
   1    4.0490   -2.9765    0.0000    5.0253   1  0  0
   1   -4.0490    2.9765    0.0000    5.0253   0  1  0
   2   -0.0000    5.9530    0.0000    5.9530   0  1  0
   2    0.0000   -5.9530    0.0000    5.9530   0 -1  0
   1   -0.8391    2.9765    7.1389    7.7799   1  1  1
   1    0.8391   -2.9765   -7.1389    7.7799   0  0 -1
   1    0.8391    2.9765   -7.1389    7.7799   0  1 -1
   1   -0.8391   -2.9765    7.1389    7.7799   1  0  1
   2    4.8881    0.0000   -7.1389    8.6520   0  0 -1
   2   -4.8881   -0.0000    7.1389    8.6520   0  0  1
   2   -4.8881    5.9530    7.1389   10.5022   0  1  1
   2    4.8881   -5.9530   -7.1389   10.5022   0 -1 -1
[data]$
```

The first 2 lines are comments. The 3rd line is the number of molecules in unit cell. Then follows the information with each molecule in the unit cell. The neighbor Center Of Mass (COM) *dx*, *dy*, *dz*, and *r* are relative to the corresponding central root molecule COM in the unit cell. The `cindex` stands for cell index, indicating which unit cell the neighbor molecule is located in.

These files can be visualized by some graphic tools. For example, one can use Mercury program (https://www.ccdc.cam.ac.uk/support-and-resources/Downloads/) to open the file `mol1_neighbors_mid.cif`, tick "Show Labels for", select "**Non C, H**" with "**Atom Label**" , one can obtain the following figure:





The cross（x） indicates the COM position, the number is the neighbor molecular ID, one can even measure the COM distance with mouse click.

The files `mol*_bonds.dat` are used internally with `evc`.

The reorganization energies obtained by using `evc` are put in `mol_lambda_evc-e.dat` and `mol_lambda_evc-h.dat`, while the reorganization energies by the four-point method are put in `mol_lambda_4P-e.dat` and `mol_lambda_4P-h.dat`. One should compare these two sets of data, if they are close, then the calculated reorganization energies are normally reliable.

The transfer integrals are put in files `VH.dat` and `VL.dat`, the `VH.dat` is shown as follows:

```
[data]$ cat VH.dat
2
14
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2
   -17.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 1 1
   -17.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 1 1
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2
    -0.076 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 1 1
    -0.076 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 1 1
    -0.001 meV  # 2mol-13.log uc_mol-1.log nei_mol-13.log 1 1
    -0.001 meV  # 2mol-13.log uc_mol-1.log nei_mol-13.log 1 1
14
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
    -1.570 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
   -17.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 2 2
   -17.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 2 2
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
    -5.734 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
    -0.076 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 2 2
    -0.076 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 2 2
    -0.001 meV  # 2mol-13.log uc_mol-1.log nei_mol-13.log 2 2
    -0.001 meV  # 2mol-13.log uc_mol-1.log nei_mol-13.log 2 2
[data]$
```

The first line shows the number of molecules in the unit cell, then follows the number of neighbors and the corresponding transfer integrals for each central molecule in the unit cell. Comments are starting with #. The last two columns show the molecular ID in the unit cell.

One can plot transfer integrals plus COMs with script $MOMAP_ROOT/scripts/plot-transint-V.py under the `data` directory, it will create several .pov files, e.g.,

```
$ $MOMAP_ROOT/scripts/plot-transint-V.py
```

```
[data]$ ls *.pov
mol-01-VH.pov mol-01-VL.pov mol-02-VH.pov mol-02-VL.pov
[data]$
```

These are simple .pov files, one can use `povray` program to render them to high quality .png files, e.g.,

```
$ povray mol-01-VH.pov
```

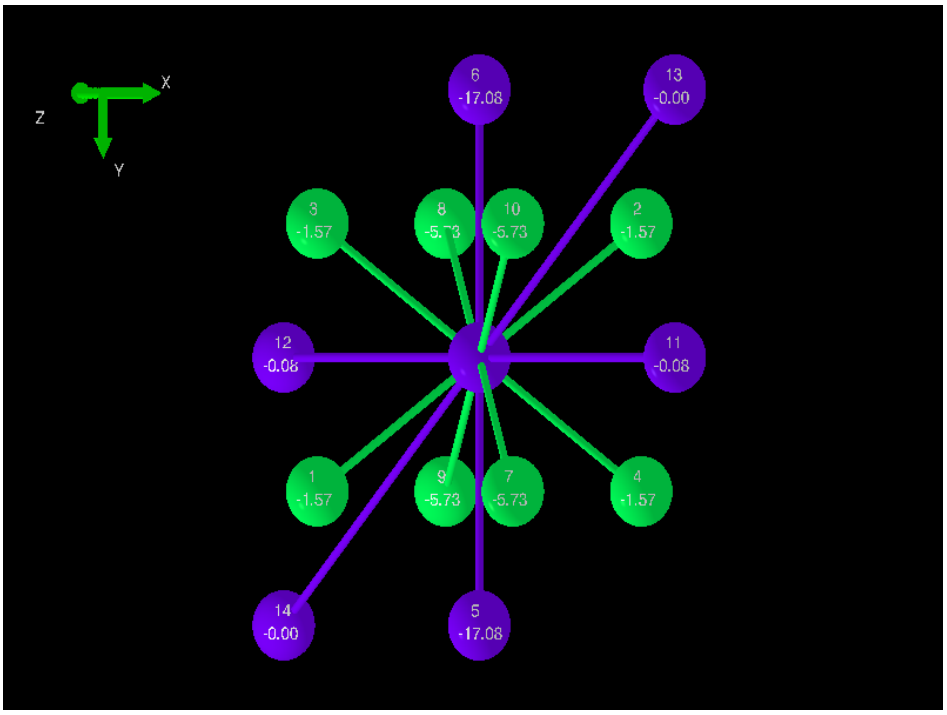


Figure showing neighbor molecular ID plus transfer integral value

INFORMATION UNDER MC-QUANTUM/MARCUS

Depending on the option `chargetype` in `momap.inp`, it can be `elec`, `hole`, or `all`. Here, we take `elec` option as an example.

The files in `elec` directory is shown as follows:

```
[elec]$ ls
NM-e.dat                  mob_a.dat                 mob_plane_xy.gnu-png      mob_plane_yz.py
VL.dat                    mob_b.dat                 mob_plane_xy.py           momap.inp
WL.dat                    mob_c.dat                 mob_plane_xz.dat          naphthalene.cif
charge.inp                mob_direction_all.dat     mob_plane_xz.gnu          prepare-mc.py
get_mob.local             mob_direction_all.gnu     mob_plane_xz.gnu-png      run_mc.local
get_mob.pbs               mob_direction_all.gnu-png mob_plane_xz.py           run_mc.pbs
get_mob.py                mob_direction_all.py      mob_plane_yz.dat          run_mc_batch.py
hoprates.dat              mob_plane_xy.dat          mob_plane_yz.gnu          track
mob.dat                   mob_plane_xy.gnu          mob_plane_yz.gnu-png      w0.dat
[elec]$
```

The NM-e.dat is obtained by using `evc.exe`. The `hoprates.dat` is the hopping rate file for charge carrier electron, as shown below:

```
[elec]$ cat hoprates.dat
# nroot, root_mol_info, nnei, neighbors_info
# r = molecular COM distance, cindex = neighbor cell index
2
1  -0.0000   0.0000   0.0000  # root
  14 #     dx        dy        dz         r    cindex        transint            hoprate
   2   -4.0490   -2.9765   -0.0000    5.0253  -1 -1  0    2.669500e+01 meV    5.533090e+03 1/ns
   2    4.0490    2.9765   -0.0000    5.0253   0  0  0    2.669500e+01 meV    5.533090e+03 1/ns
   2   -4.0490    2.9765   -0.0000    5.0253  -1  0  0    2.669500e+01 meV    5.533090e+03 1/ns
   2    4.0490   -2.9765   -0.0000    5.0253   0 -1  0    2.669500e+01 meV    5.533090e+03 1/ns
   1    0.0000   -5.9530    0.0000    5.9530   0 -1  0    8.172000e+00 meV    5.185189e+02 1/ns
   1   -0.0000    5.9530    0.0000    5.9530   0  1  0    8.172000e+00 meV    5.185189e+02 1/ns
   2    0.8391   -2.9765   -7.1389    7.7799  -1 -1 -1    1.244000e+00 meV    1.201569e+01 1/ns
   2   -0.8391    2.9765    7.1389    7.7799   0  0  1    1.244000e+00 meV    1.201569e+01 1/ns
   2   -0.8391   -2.9765    7.1389    7.7799   0 -1  1    1.244000e+00 meV    1.201569e+01 1/ns
   2    0.8391    2.9765   -7.1389    7.7799  -1  0 -1    1.244000e+00 meV    1.201569e+01 1/ns
   1    4.8881    0.0000   -7.1389    8.6520   0  0 -1   -2.119000e+00 meV    3.486340e+01 1/ns
   1   -4.8881   -0.0000    7.1389    8.6520   0  0  1   -2.119000e+00 meV    3.486340e+01 1/ns
   1    4.8881    5.9530   -7.1389   10.5022   0  1 -1    1.000000e-03 meV    7.764399e-06 1/ns
   1   -4.8881   -5.9530    7.1389   10.5022   0 -1  1    1.000000e-03 meV    7.764399e-06 1/ns
2   4.0490   2.9765  -0.0000  # root
  14 #     dx        dy        dz         r    cindex        transint            hoprate
   1    4.0490    2.9765    0.0000    5.0253   1  1  0    2.669500e+01 meV    5.533090e+03 1/ns
   1   -4.0490   -2.9765    0.0000    5.0253   0  0  0    2.669500e+01 meV    5.533090e+03 1/ns
   1    4.0490   -2.9765    0.0000    5.0253   1  0  0    2.669500e+01 meV    5.533090e+03 1/ns
   1   -4.0490    2.9765    0.0000    5.0253   0  1  0    2.669500e+01 meV    5.533090e+03 1/ns
   2   -0.0000    5.9530    0.0000    5.9530   0  1  0    8.172000e+00 meV    5.185189e+02 1/ns
   2    0.0000   -5.9530    0.0000    5.9530   0 -1  0    8.172000e+00 meV    5.185189e+02 1/ns
   1   -0.8391    2.9765    7.1389    7.7799   1  1  1    1.244000e+00 meV    1.201569e+01 1/ns
   1    0.8391   -2.9765   -7.1389    7.7799   0  0 -1    1.244000e+00 meV    1.201569e+01 1/ns
   1    0.8391    2.9765   -7.1389    7.7799   0  1 -1    1.244000e+00 meV    1.201569e+01 1/ns
   1   -0.8391   -2.9765    7.1389    7.7799   1  0  1    1.244000e+00 meV    1.201569e+01 1/ns
   2    4.8881    0.0000   -7.1389    8.6520   0  0 -1   -2.119000e+00 meV    3.486340e+01 1/ns
   2   -4.8881   -0.0000    7.1389    8.6520   0  0  1   -2.119000e+00 meV    3.486340e+01 1/ns
   2   -4.8881    5.9530    7.1389   10.5022   0  1  1    1.000000e-03 meV    7.764399e-06 1/ns
   2    4.8881   -5.9530   -7.1389   10.5022   0 -1 -1    1.000000e-03 meV    7.764399e-06 1/ns
[elec]$
```

The file format is similar to that of `neighbor.dat`, but with extra columns.

The mobilities are put in files `mob_*.dat`. The handy plot scripts are generated for Gnuplot and Python. If the Gnuplot version is 5.0 or above, one can use the `*.gnu-png` scripts to generate the .png files directly, e.g.,

```
$ gnuplot *.gnu-png
```

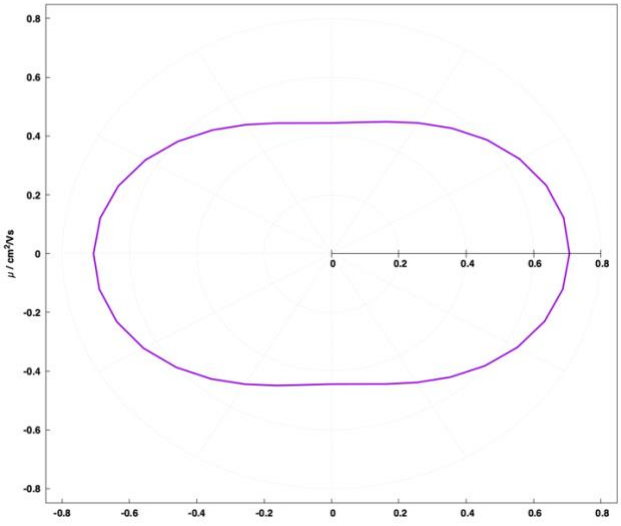
Then, four .png files are generated as follows:

```
[elec]$ ls *.png
mob_direction_all.png mob_plane_xy.png      mob_plane_xz.png      mob_plane_yz.png
[elec]$
```

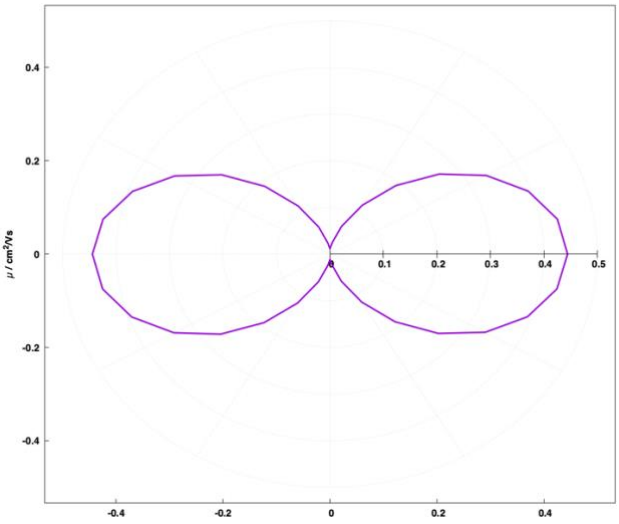
In Linux system, one can visualize the files by using `display` program, e.g.,

```
$ display *.png
```

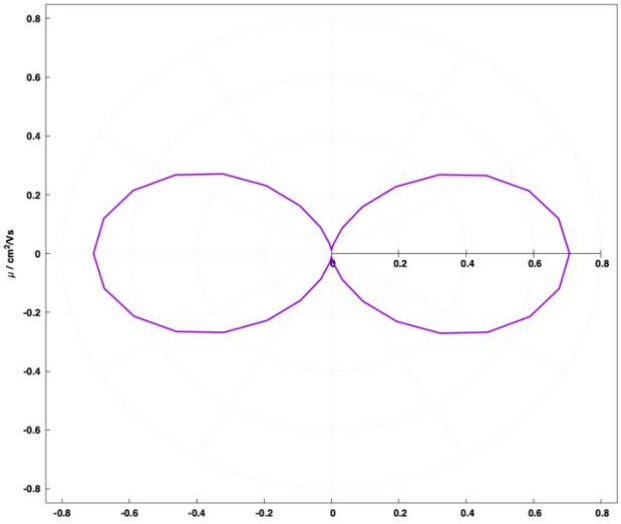
Hit the spacebar to view the next .png file if available.



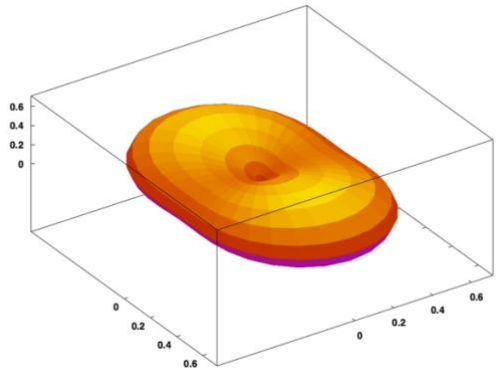
xy plane



yz plane



xz plane



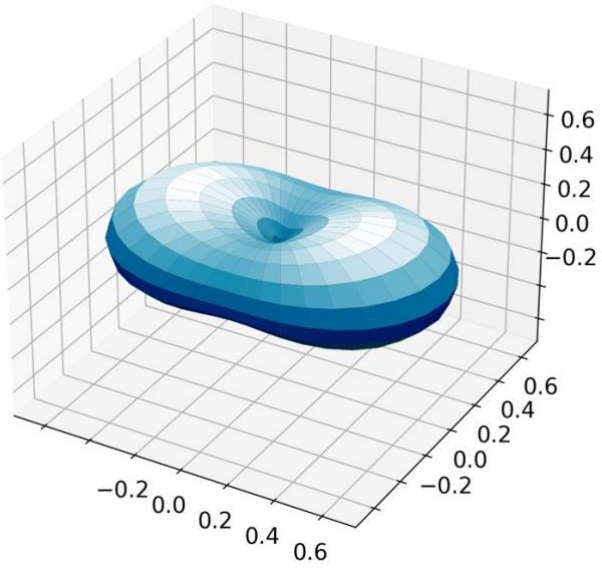
3D plot

If the `numpy` and `matplotlib` are installed with Python, one use the Python scripts to plot the figures as well.

```
[elec]$ ls mob*.py
mob_direction_all.py mob_plane_xy.py      mob_plane_xz.py      mob_plane_yz.py
[elec]$
```

As an example, one can visualize the 3D plot with the following command:

```
$ python mob_direction_all.py
```



3D Plot

The other script files can be run similarly.

GATHERING OF OUTPUT DATA

The output data are collected and assembled in file `momap-quantum.dat` or `momap-marcus.dat`, as shown below:

```
[naphthalene]$ cat momap-quantum.dat
  __  __    ___    __  __      _      ____
 |  \/  |  / _ \  |  \/  |    / \    |  _ \
 | |\/| | | | | | | |\/| |   / _ \   | |_) |
 | |  | | | |_| | | |  | |  / ___ \  |  __/
 |_|  |_|  \___/  |_|  |_| /_/   \_\ |_|       Version 2021A (2.3.1)

 Copyright (c) 2017 Shuaigroup @ Tsinghua University &
         Institute of Chemistry, Chinese Academy of Sciences.
         All Rights Reserved.

Cite transport work as (required):
1. Y. Niu, W. Li, Q. Peng, H. Geng, Y. Yi, L. Wang, G. Nan, D. Wang, Z. Shuai,
   Molecular Physics, 2018, doi: 10.1080/00268976.2017.1402966.
2. G. Nan, X. Yang, L. Wang, Z. Shuai, Y. Zhao, Phys. Rev. B, 2009, 79, 115203.
3. Z. Shuai, D. Wang, Q. Peng, H. Geng, Acc. Chem. Res., 2014, 47, 3301.

Cite transport work as (recommended):
1. Z. Shuai, H. Geng, W. Xu, Y. Liao, J-M. Andre, Chem. Soc. Rev., 2014, 43, 2662.

Running configuration:
  data/config.inp

Molecular information:
  data/mol1.mol
  data/mol2.mol

Neighbor information:
  data/neighbor.dat
  data/mol1_neighbors.cif
  data/mol2_neighbors.cif

Transfer integral V:
  data/VH.dat
  data/VL.dat
```

Continue to the next page

```
Reorganization energies by evc:
  data/mol_lambda-e.dat
  data/mol_lambda-h.dat

Reorganization energies by four point method:
  data/mol_lambda_4P-e.dat
  data/mol_lambda_4P-h.dat

EVC NM information:
  data/NM-e.dat
  data/NM-h.dat

**** Hopping rates for MC-quantum/elec:
  MC-quantum/elec/hoprates.dat

**** Mobility data for MC-quantum/elec
mob_a / dev [cm**2/Vs]:   7.067352e-01    5.081837e-02
mob_b / dev [cm**2/Vs]:   4.442726e-01    3.402536e-02
mob_c / dev [cm**2/Vs]:   2.359066e-01    2.805255e-02
mob_ave / dev [cm**2/Vs]:   3.876372e-01    2.605006e-02
Directional mobilities are in file:
  MC-quantum/elec/mob_direction_all.dat
**** End of Mobility data for MC-quantum/elec

**** Hopping rates for MC-quantum/hole:
  MC-quantum/hole/hoprates.dat

**** Mobility data for MC-quantum/hole
mob_a / dev [cm**2/Vs]:   3.781213e-03    2.733651e-04
mob_b / dev [cm**2/Vs]:   3.259527e-01    3.323223e-02
mob_c / dev [cm**2/Vs]:   8.082982e-02    6.452564e-03
mob_ave / dev [cm**2/Vs]:   1.434958e-01    7.956009e-03
Directional mobilities are in file:
  MC-quantum/hole/mob_direction_all.dat
**** End of Mobility data for MC-quantum/hole

Normal end of MOMAP data gathering.
```

The file is pretty self-explanatory, however, one should also check the log file for any abnormality.

ERROR CHECKING

One should check the log file carefully for any abnormality. If an error occurs, one should check the log file and `RUN` directory to locate where the error comes from, and then fix it accordingly.