MOMAP

Tutorial 09

Mobility Calculation by Considering Dynamic Disorder of Transfer Integral

Version 2020B

May, 2021

MOMAP Tutorial 09

Version 2020B edited by:

Dr. Xingliang Peng

Dr. Qikai Li

Released by Hongzhiwei Technology (Shanghai) Co., Ltd

and Z.G. Shuai Group

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

TABLE OF CONTENTS

MOMAP Tutorial

Mobility Calculation by Considering Dynamic Disorder of Transfer Integral

BACKGROUND

In organic semiconductor, molecules are packed up by weak van der Waals interaction, which tends to result in large intermolecular displacements. The intermolecular displacement caused by thermal motion consequently leads to fluctuation or so-called dynamic disorder in the intermolecular transfer integral, which could be in the same order of magnitude as the transfer integral itself, and thus cannot be ignored. As a result, in these situations, we should also include this part in the calculation of mobilities.[1, 2]

Reference:

1. L. Wang, Q. Li, Z. Shuai, L. Chen and Q. Shi, *Multiscale study of charge mobility of organic semiconductor with dynamic disorders, Phys Chem Chem Phys*, 2010, 12, 3309-3314.
2. X. Peng, Q. Li and Z. Shuai, *Influences of dynamic and static disorder on the carrier mobility of BTBT-C12 derivatives: a multiscale computational study, Nanoscale*, 2021, 13, 3252-3262.

QUICK START

Different from the normal mobility calculation in MOMAP (see tutorials 04 and 08), we need the extra information about the fluctuation of the intermolecular transfer integral, which is aroused from the fluctuation of relative position between molecules. This fluctuation can be obtained by doing a simple molecular dynamics (MD) simulation. The fluctuant transfer integral can be included in the KMC process, and finally we can obtain the mobility as usual.

1. MD simulation

We can perform the MD simulations to obtain the fluctuation of intermolecular displacement at ambient environment, and then calculate the fluctuation of transfer integral based on the MD stacking. Here, we show a MD example of pentacene (GAFF force field is used) by using the popular GROMACS package.

Before carrying out the MD simulation, we need to calculate the RESP charge and then prepare the GROMACS topology file .top and coordinate file .gro first. In this step, we use `antechamber` (a program with `amber`, and one can download the `ambertools` from the website at http://ambermd.org/antechamber/download.html) and `acpype.py` program from website at https://pypi.org/project/acpype/.

1a) Fist we need to convert the crystal file **penta.cif** to Gaussian input file, and modify the key words accordingly in order to perform the structural optimization at the B3LYP/6-31G* level.

By using the following MOMAP tool, we can obtain two .com files (**penta-1.com** and **penta-2.com**). Rename **penta-1.com** to **penta-opt.com**, and then modify the key words as shown below in order to do the structural optimization.

```
$ momap_cif2gjf.exe penta.cif
```

```
$ cat penta.cif
_symmetry_cell_setting              triclinic
_symmetry_space_group_name_H-M      'P -1'
_symmetry_int_tables_number         2
_symmetry_space_group_name_Hall     '-P 1'
loop_
_symmetry_equiv_pos_as_xyz
x,y,z
-x,-y,-z
_cell_length_a                  5.958
_cell_length_b                  7.596
_cell_length_c                  15.6096
```

```
_cell_angle_alpha                81.25
_cell_angle_beta                 86.56
_cell_angle_gamma                89.80
_cell_volume                     696.953
_cell_formula_units_Z            2
loop_
_atom_site_label
_atom_site_type_symbol
_atom_site_fract_x
_atom_site_fract_y
_atom_site_fract_z
_atom_site_U_iso_or_equiv
_atom_site_thermal_displace_type
_atom_site_occupancy
C1 C  0.2107   -0.0830  -0.0070  0.0373 Uiso 1.0000
C2 C  0.1185   -0.0666    0.0758 0.0373 Uiso 1.0000
C3 C  0.2245   -0.1278    0.1522 0.0373 Uiso 1.0000
C4 C  0.1271   -0.1085    0.2334 0.0373 Uiso 1.0000
C5 C  0.2409   -0.1703    0.3115 0.0373 Uiso 1.0000
C6 C  0.1479   -0.1506    0.3900 0.0373 Uiso 1.0000
C7 C  -0.0693 -0.0675    0.3979 0.0373 Uiso 1.0000
C8 C  -0.1802 -0.0072    0.3255 0.0373 Uiso 1.0000
C9  C -0.0875 -0.0232    0.2397 0.0373 Uiso 1.0000
C10 C -0.1947 0.0375  0.1660    0.0373 Uiso 1.0000
C11 C -0.0998 0.0189  0.0830    0.0373 Uiso 1.0000
C12 C 0.7104  0.5869 -0.0091    0.0373 Uiso 1.0000
C13 C 0.6200  0.5248  0.0746    0.0373 Uiso 1.0000
C14 C 0.7271  0.5446  0.1499    0.0373 Uiso 1.0000
C15 C 0.6310  0.4811  0.2322    0.0373 Uiso 1.0000
C16 C 0.7428  0.5052  0.3092    0.0373 Uiso 1.0000
C17 C 0.6495  0.4452  0.3887    0.0373 Uiso 1.0000
C18 C 0.4349  0.3541  0.3987    0.0373 Uiso 1.0000
C19 C 0.3250  0.3300  0.3274    0.0373 Uiso 1.0000
C20 C 0.4160  0.3932  0.2406    0.0373 Uiso 1.0000
C21 C 0.3082  0.3718  0.1680    0.0373 Uiso 1.0000
C22 C 0.4025  0.4345  0.0840    0.0373 Uiso 1.0000
H1  H 0.3496  -0.1379  -0.0122  0.0373 Uiso 1.0000
H3  H 0.3636  -0.1829  0.1485   0.0373 Uiso 1.0000
H5  H 0.3805  -0.2244  0.3073   0.0373 Uiso 1.0000
H6  H 0.2238  -0.1907  0.4395   0.0373 Uiso 1.0000
H7  H -0.1335 -0.0553  0.4525   0.0373 Uiso 1.0000
H8  H -0.3199 0.0459   0.3317   0.0373 Uiso 1.0000
H10 H -0.3338 0.0923   0.1708   0.0373 Uiso 1.0000
H12 H 0.8496  0.6441  -0.0158   0.0373 Uiso 1.0000
H14 H 0.8662  0.6016   0.1448   0.0373 Uiso 1.0000
H16 H 0.8812  0.5633   0.3036   0.0373 Uiso 1.0000
H17 H 0.7235  0.4624   0.4376   0.0373 Uiso 1.0000
H18 H 0.3719  0.3118   0.4540   0.0373 Uiso 1.0000
H19 H 0.1870  0.2710   0.3349   0.0373 Uiso 1.0000
H21 H 0.1692  0.3145   0.1742   0.0373 Uiso 1.0000
```

```
$ cat penta-opt.com
…
#P B3LYP 6-31g* nosymm

penta-1

0 1
…
```

1b) After finishing the optimization with Gaussian `g09` or `g16`, we proceed to do the calculation of RESP charge at the `HF/6-31G**` level by using the optimized structure from pervious step, and obtain the log file `penta-resp.log`. The typical gausssian input key words for RESP charge fit are shown below.

```
$ cat penta-resp.com
   …
   #P hf 6-31g** nosymm pop=mk iop(6/33=2,6/41=10,6/42=17)

   sp

   0 1
   …
```

1c) With the Gaussian output from previous step, we can now fit the RESP charge by using the amber tool `antechamber`. By running the following command, we can obtain the `penta.mol2` (containing the RESP charge and atomic type information).

```
$ antechamber -i penta-resp.log -fi gout -o penta.mol2 -fo mol2 -c resp -cf esp.mol2 -s 2

$ cat penta.mol2
…
   @<TRIPOS>ATOM
        1 C1          2.2100    9.3200   15.2920 ca         1 MOL     -0.265591
        2 C2          1.7380    9.6410   16.5670 ca         1 MOL      0.098685
        3 C3          2.4400    9.3570   17.7440 ca         1 MOL     -0.368098
   …
@<TRIPOS>BOND
    1    1    2 ar
    2    1   12 1
    3    1   29 ar
    …
```

As to the meaning of various parameters, please refer to the AmberTool manual.

1d) The next step is to generate the Gromacs .top file based on **penta.mol2** file
Copy and rename **penta.mol2** to **1.mol2**.
Prepare a **tmp** file containing the following contents:

```
$ cat tmp
   source leaprc.gaff
   mods=loadamberparams 1.frcmod
   MOL=loadmol2 1.mol2
   check MOL
   saveamberparm MOL 1.prmtop 1.inpcrd
   Quit
```

By running the following commands, we can obtain **MOL_GMX.top** file.

```
$ parmchk -i 1.mol2 -f mol2 -o 1.frcmod
$ tleap -f tmp
$ acpype.py -p 1.prmtop -x 1.inpcrd -d
```

1e) Now we create a 5 × 4 × 2 supercell based on `penta.cif` file, and convert the obtained `penta542.cif` to `penta542.gro` as shown below. We need also to alter the molecule number in `penta.top` file to `80`, which corresponds to the molecule number in `penta542.gro`.

```
$ momap_cif_expand.exe penta.cif penta542.cif -nx 5 -ny 4 -nz 2

$ momap_cif2pdb.exe penta542.cif penta542.pdb

$ editconf -f penta542.pdb -o penta542.gro
```

```
$ cat penta.top
…
[ molecules ]
; Compound      nmols
 MOL             80
```

1f) Modify the atomic name in `penta.top` to match the atomic name in `penta542.gro`.

Atomic name in the original `penta.top`:

```
…
[ atoms ]
;   nr  type  resi  res  atom  cgnr     charge      mass    ; qtot bond_type
    1   ca     1   MOL    C1    1    -0.265601    12.01000 ; qtot -0.266
    2   ca     1   MOL    C2    2     0.098692    12.01000 ; qtot -0.167
…
   10   ca     1   MOL   C10   10    -0.368097   12.01000 ; qtot -1.385
   11   ca     1   MOL   C11   11     0.098692   12.01000 ; qtot -1.286
…
   35   ha     1   MOL   H13   35     0.187969    1.00800 ; qtot -0.212
   36   ha     1   MOL   H14   36     0.211648    1.00800 ; qtot -0.000
…
```

The atomic name in `penta542.gro`:

```
momap
 2880
```

```
    1MOL     C1    1   0.221   0.932   1.529
    1MOL     C2    2   0.174   0.964   1.657
…
    1MOL     CA   10  -0.004   1.065   1.796
    1MOL     CB   11   0.045   1.031   1.668
…
    1MOL    H23   35   0.256   0.883   1.029
     1MOL   H24   36   0.279   0.886   1.277
…
```

Atomic name in the modified “penta.top”:

```
…
[ atoms ]
;   nr  type  resi  res  atom  cgnr   charge      mass       ; qtot bond_type
     1   ca     1   MOL    C1    1   -0.265601    12.01000  ; qtot -0.266
     2   ca     1   MOL    C2    2    0.098692    12.01000  ; qtot -0.167
…
    10   ca     1   MOL    CA   10   -0.368097    12.01000 ; qtot -1.385
    11   ca     1   MOL    CB   11    0.098692    12.01000 ; qtot -1.286
…
    35   ha     1   MOL   H23   35    0.187969     1.00800 ; qtot -0.212
    36   ha     1   MOL   H24   36    0.211648     1.00800 ; qtot -0.000
…
```

1g) Before we continue to carry out the MD simulations, we’d better first do a calculation of energy minimum (EM).

A typical gromcas parameter setting file `em.mdp` is shown as follows:

```
define          = -DFLEXIBLE
integrator      = cg
nsteps          = 3000
emtol           = 10.0
emstep          = 0.01
;
nstxout         = 200
nstlog          = 200
nstenergy       = 200
;
pbc             = xyz
cutoff-scheme   = Verlet
nstlist         = 1
ns-type         = grid
coulombtype     = PME
rcoulomb        = 1.2
vdwtype         = Cut-off
rvdw            = 1.2
DispCorr        = EnerPres
;
constraints     = none
```

We then use Gromacs `grompp` to combine all the files together by executing the following command to generate the final EM run file `em.tpr`:

```
gmx grompp -f em.mdp -c penta542.gro -p penta.top -o em.tpr
```

If everything is fine, we then obtained an `em.tpr` file. Now we do an EM calculation by using the following command:

```
gmx mdrun -v -deffnm em
```

Once we obtain the `em.gro` file, we use it as our initial MD structure, similarly, we can generate the final MD run file `md.tpr`.

```
gmx grompp -f eq.mdp -c em.gro -p penta.top -o md.tpr
```

Again, if everything is fine, we then obtained the md.tpr file. Now we do a MD calculation by using the following command:

```
gmx mdrun -v -deffnm md
```

After finishing the MD run, we extract 2000 snapshots every 30 fs after thermal equilibration, and then do an EM run for each snapshot. For each snapshot, we then calculate the transfer integral of each typical dimer (such as dimer A, B, C in the following figure).

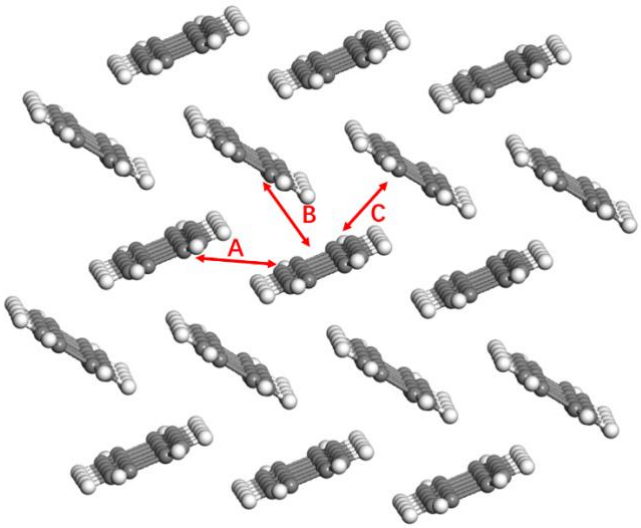

Figure 1 transfer integral calculation for each typical dimer

This step can be very tedious, we directly use the QC program (e.g., Gaussian g09/g16) to calculate the transfer integrals for the typical dimers with all the snapshots, and then assemble the data into files. Here, we have only calculated the important dimers, that is, with larger transfer integrals, to simplify the preparation.

2. Calculate mobility

As we have the dynamic transfer integral of each typical dimer in a period of time, and we can now do a discrete Fourier transformation to get the time-dependent transfer integral fluctuation.

$$V_{mn}(t) = \langle V_{mn} \rangle + \sum_{k=1}^{N-1} ReV_k cos(\omega t + \varphi_0) + \sum_{k=1}^{N-1} ImV_k sin(\omega t + \varphi_0)$$

Firstly, we do a normal transport mobility calculation based on the crystal file `penta.cif`.

```
$ 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                = 0
     do_transport_get_mob_MC            = 0
     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     = 0

     # Job Scheduling
     sched_type        = slurm    ! pbs, slurm, lsf, or local
     queue_name        = X32Cv4

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

     qc_method         = zindo
     qc_basis          = zindo
     qc_basis_re       = b3lyp 6-31g*
     qc_memory         = 4096     ! MB
     qc_nodes          = 1
     qc_ppn            = 32
     lat_cutoff        = 4        ! for neighbor list construction
     crystal           = penta.cif
   /
```

After finishing the calculation, we can get `data/VH.dat` file, which contains the transfer integral of all neighbors of each molecule in crystal.

In order to calculate the mobility with dynamic disorder, we need to prepare the dynamic transfer integral file `VH?-dyn.dat` for hole transport or `VL?-dyn.dat` for electron transport. As there exist two molecules in the crystal, we need to prepare two dynamic transfer integral file `VH1-dyn.dat` for the first molecule and `VH2-dyn.dat` for the second molecule.

In the dynamic transfer integral file, the first line is the comment line and will be skipped by the reading program. The second line has 3 numbers, which correspond to the snapshot number, the

neighbor number and interval time (unit in fs) between snapshots respectively. The rest lines are the transfer integral of each neighbor in each snapshot.

```
$ cat data/VH.dat
   2
   12
       65.170 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2    neighbor1
       65.170 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 1 2    neighbor2
       60.923 meV  # 2mol-3.log uc_mol-1.log nei_mol-3.log 1 2    neighbor3
       60.923 meV  # 2mol-3.log uc_mol-1.log nei_mol-3.log 1 2    neighbor4
      -56.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 1 1    neighbor5
      -56.077 meV  # 2mol-5.log uc_mol-1.log nei_mol-5.log 1 1    neighbor6
       -0.442 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2    neighbor7
       -0.442 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 1 2    neighbor8
        0.229 meV  # 2mol-9.log uc_mol-1.log nei_mol-9.log 1 1    neighbor9
        0.229 meV  # 2mol-9.log uc_mol-1.log nei_mol-9.log 1 1    neighbor10
       -0.581 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 1 2   neighbor11
       -0.581 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 1 2   neighbor12
   12
       65.170 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
       65.170 meV  # 2mol-1.log uc_mol-1.log nei_mol-1.log 2 1
       60.923 meV  # 2mol-3.log uc_mol-1.log nei_mol-3.log 2 1
       60.923 meV  # 2mol-3.log uc_mol-1.log nei_mol-3.log 2 1
      -54.608 meV  # 2mol-17.log uc_mol-2.log nei_mol-17.log 2 2
      -54.608 meV  # 2mol-17.log uc_mol-2.log nei_mol-17.log 2 2
       -0.442 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
       -0.442 meV  # 2mol-7.log uc_mol-1.log nei_mol-7.log 2 1
        1.575 meV  # 2mol-21.log uc_mol-2.log nei_mol-21.log 2 2
        1.575 meV  # 2mol-21.log uc_mol-2.log nei_mol-21.log 2 2
       -0.581 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 2 1
       -0.581 meV  # 2mol-11.log uc_mol-1.log nei_mol-11.log 2 1
```

```
$ cat VH1-dyn.dat
   #nFrame nV dt(fs)  Comment line
   2051 12 20          snapshot_number neighor_number interval_time
   -31.618             snapshot1 neighbor1
   -31.618             snapshot1 neighbor2
   -65.154             snapshot1 neighbor3
   -65.154             snapshot1 neighbor4
   -54.608             snapshot1 neighbor5
   -54.608             snapshot1 neighbor6
    -0.442             snapshot1 neighbor7
    -0.442             snapshot1 neighbor8
     1.575             snapshot1 neighbor9
     1.575             snapshot1 neighbor10
    -0.581             snapshot1 neighbor11
    -0.581             snapshot1 neighbor12
   -35.646             snapshot2 neighbor1
   -35.646             snapshot2 neighbor2
   -69.800             snapshot2 neighbor3
   …                 …
```

After preparing the dynamic transfer integral files, we need to put them in the `data` folder, and add `V_dynamic_disorder = 1` in the control file `momap.inp`.

Then, we can run the KMC based on the dynamic transfer integrals, and then obtain the mobility by considering dynamic disorder of transfer integral in `momap-marcus.dat` file.

```
$cat momap.inp
   &transport
    do_transport_prepare                = 0
    do_transport_submit_HL_job          = 0
    do_transport_get_transferintegral   = 0
    do_transport_submit_RE_job          = 0
    do_transport_get_re_evc             = 0
    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            = slurm    ! pbs, slurm, lsf, or local
    queue_name            = X32Cv4

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

    qc_method             = zindo
    qc_basis              = zindo
    qc_basis_re           = b3lyp 6-31g*
    qc_memory             = 4096     ! MB
    qc_nodes              = 1
    qc_ppn                = 32

    temp                  = 300
    chargetype            = h
    ratetype              = marcus   ! marcus or quantu
    lat_cutoff            = 4        ! for neighbor list construction
    V_dynamic_disorder    = 1

    nsimu                 = 2000
    tsimu                 = 100      ! in ns
    tsnap                 = 0.5

    crystal               = penta.cif
   /
```

Finally, we have done with the mobility calculation by considering dynamic disorder of transfer integral.