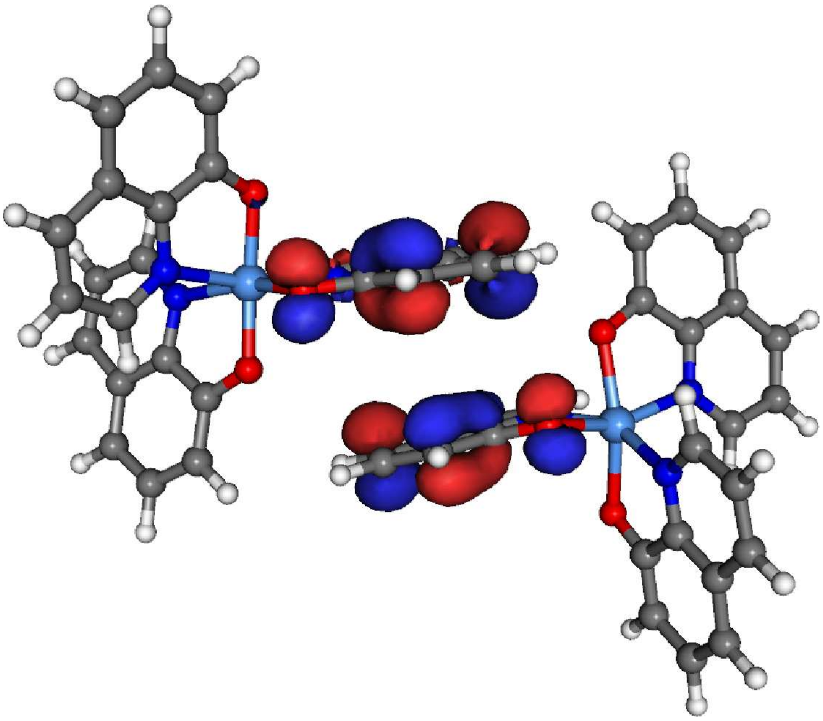
VOTCA-CTP

Charge Transport Simulations

User Manual



compiled from: 1.6-dev (gitid: c05ceee)

June 21, 2019

www.votca.org

Disclamer

Open source projects are made available and contributed to under licenses that include terms that, for the protection of contributors, make clear that the projects are offered "as-is", without warranty, and disclaiming liability for damages resulting from using the projects.

Citations

Development of this software depends on academic research grants. If you are using the package, please cite the following papers

[1] Long-range embedding of molecular ions and excitations in a polarizable molecular environment, Carl Poelking and Denis Andrienko
J. Chem. Theory Comp. 12, 4516, 2016

[2] Modeling of spatially correlated energetic disorder in organic semiconductors, Pascal Kordt, Denis Andrienko
J. Chem. Theory Comput., 12, 36, 2016

[3] Microscopic simulations of charge transport in disordered organic semiconductors, Victor Rühle, Alexander Lukyanov, Falk May, Manuel Schrader, Thorsten Vehoff, James Kirkpatrick, Björn Baumeier and Denis Andrienko
J. Chem. Theor. Comp. 7, 3335, 2011

[4] Extracting nondispersive charge carrier mobilities of organic semiconductors from simulations of small systems, A. Lukyanov, D. Andrienko
Phys. Rev. B, 82, 193202, 2010

[5] Density-functional based determination of intermolecular charge transfer properties for large-scale morphologies, Björn Baumeier, James Kirkpatrick, and Denis Andrienko
Phys. Chem. Chem. Phys. 12, 11103, 2010

[6] Versatile Object-oriented Toolkit for Coarse-graining Applications, Victor Rühle, Christoph Junghans, Alexander Lukyanov, Kurt Kremer and Denis Andrienko
J. Chem. Theor. Comp. 5, 3211, 2009

[7] An approximate method for calculating transfer integrals based on the ZINDO Hamiltonian, James Kirkpatrick,
Int. J. Quantum Chem. 108, 51, 2007

Copyright

VOTCA-CTP is free software. The entire package is available under the Apache License. For details, check the LICENSE file in the source code. The VOTCA-CTP source code is available on our homepage, www.votca.org.

Contents

Overview

Charge carrier and exciton dynamics in organic semiconductors can often be described as a sequence of charge/exciton transfer reactions between localized states. Transfer rates depend on electronic coupling elements, reorganization energies, and site energies, which vary as a function of molecular positions and orientations. The purpose of the VOTCA-CTP package is to simplify the computational workflow for charge and exciton transport simulations, which is shown in Figure 1.

In this workflow, atomistic morphology is mapped onto conjugated segments and rigid fragments. If needed, rigid fragments are substituted with the quantum-mechanically optimized copies. The conjugated segments are then used to construct a neighbor list. For each pair of this list an electronic coupling element, a reorganization energy, a driving force, and eventually the rate are evaluated.

Solid-state ionization energies, electron affinities, and excited state energies of conjugated segments are calculated perturbatively, as a sum of the gas-phase contribution and electrostatic and polarization interaction with the environment. Coulomb interactions can be evaluated using a cutoff or an aperiodic Ewald summation, available for both the bulk and slab geometries. These calculations require distributed atomic multipoles and polarizabilities for the neutral, cationic (IE), anionic (EA), or excited state.

Electronic coupling elements between conjugated segments can be performed by projecting the dimer orbitals on the respective diabatic states, approximated by the monomer orbitals. Interfaces to `Gaussian` and, to a lesser extent, to `Turbomole` are provided to perform these computationally demanding simulations. Alternatively, it is possible to use the fast molecular orbital overlap method based on the semi-empirical INDO Hamiltonian. This method requires INDO molecular orbitals in the format provided by the `Gaussian` package.

The neighbor list and rates define a directed graph. The corresponding master equation is solved using the kinetic Monte Carlo method, which allows to explicitly monitor the charge and exciton dynamics in the system as well as to calculate time- or ensemble averages of occupation probabilities, charge fluxes, correlation functions, and field-dependent mobilities.

The package is organized in several programs executing calculators. Results are stored in a sql database which is also used to restart simulations. In the following we describe individual steps required to perform charge and exciton transport simulations, the format of input and output files, and the complete reference of programs and calculators, compiled from the installed code. A tutorial is available on the github github.com/votca/ctp-tutorials.

Morphology

There is no generic recipe on how to predict a large-scale atomistically-resolved morphology of an organic semiconductor. The required methods are system-specific: for ultra-pure crystals, for example, density-functional methods can be used provided the crystal structure is known from experiment. For partially disordered organic semiconductors, however, system sizes much larger than a unit cell are required. Classical molecular dynamics or Monte Carlo techniques are then the methods of choice.

In molecular dynamics, atoms are represented by point masses which interact via empirical potentials prescribed by a force-field. Force-fields are parametrized for a limited set of compounds and their refinement is often required for new molecules. In particular, special attention shall be paid to torsion potentials between successive repeat units of conjugated polymers or between functional groups and the π-conjugated system. First-principles methods can be used to characterize the missing terms of the potential energy function.

Input files:

`conf.gro`
GROMACS trajectory

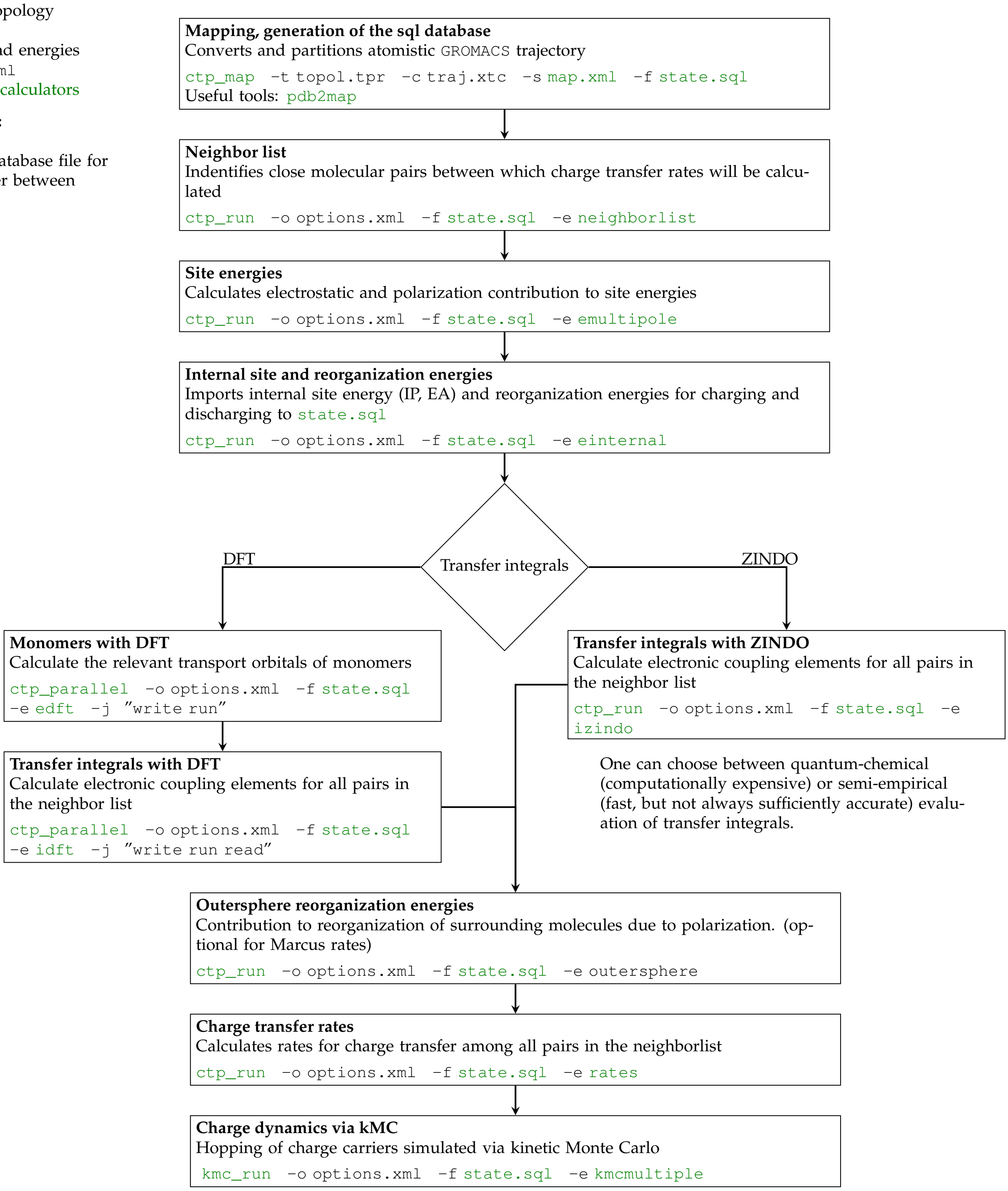
`topol.tpr`
GROMACS topology

`map.xml`
mapping and energies

`options.xml`
options for calculators

Output files:

`state.sql`
sqlite3 database file for data transfer between modules



Mapping, generation of the sql database
Converts and partitions atomistic GROMACS trajectory

```
ctp_map  -t topol.tpr  -c traj.xtc  -s map.xml  -f state.sql
```

Useful tools: `pdb2map`

Neighbor list
Indentifies close molecular pairs between which charge transfer rates will be calculated

```
ctp_run  -o options.xml  -f state.sql  -e neighborlist
```

Site energies
Calculates electrostatic and polarization contribution to site energies

```
ctp_run  -o options.xml  -f state.sql  -e emultipole
```

Internal site and reorganization energies
Imports internal site energy (IP, EA) and reorganization energies for charging and discharging to `state.sql`

```
ctp_run  -o options.xml  -f state.sql  -e einternal
```

Monomers with DFT
Calculate the relevant transport orbitals of monomers

```
ctp_parallel  -o options.xml  -f state.sql
-e edft  -j "write run"
```

Transfer integrals with DFT
Calculate electronic coupling elements for all pairs in the neighbor list

```
ctp_parallel  -o options.xml  -f state.sql
-e idft  -j "write run read"
```

Transfer integrals with ZINDO
Calculate electronic coupling elements for all pairs in the neighbor list

```
ctp_run  -o options.xml  -f state.sql  -e
izindo
```

One can choose between quantum-chemical (computationally expensive) or semi-empirical (fast, but not always sufficiently accurate) evaluation of transfer integrals.

Outersphere reorganization energies
Contribution to reorganization of surrounding molecules due to polarization. (optional for Marcus rates)

```
ctp_run  -o options.xml  -f state.sql  -e outersphere
```

Charge transfer rates
Calculates rates for charge transfer among all pairs in the neighborlist

```
ctp_run  -o options.xml  -f state.sql  -e rates
```

Charge dynamics via kMC
Hopping of charge carriers simulated via kinetic Monte Carlo

```
kmc_run  -o options.xml  -f state.sql  -e kmcmultiple
```

Get list of available calculators: `ctp_run/ctp_parallel/kmc_run  -l`
Get help and list of options for a calculator: `ctp_run/ctp_parallel/kmc_run  -d neighborlist`

Figure 1: A cheat sheet for charge and exciton transport simulations.

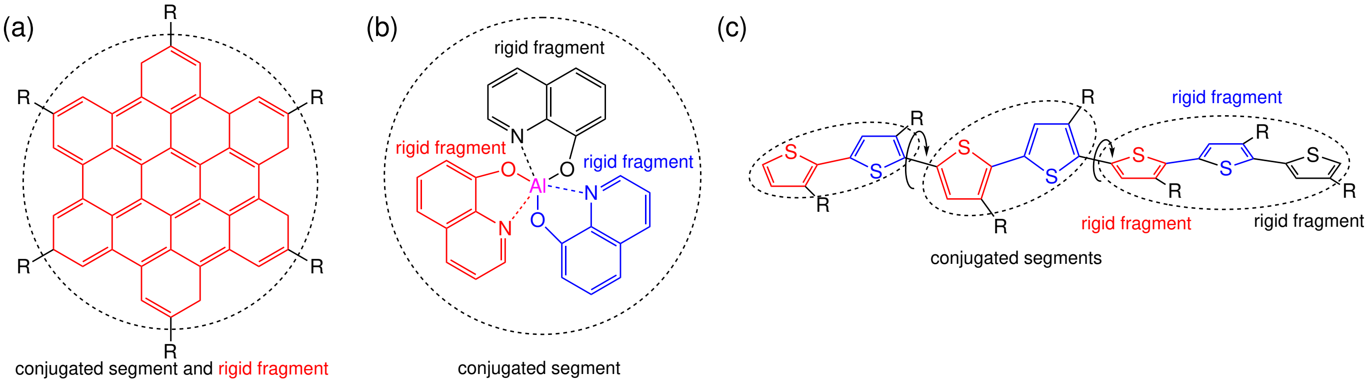


Figure 2: The concept of conjugated segments and rigid fragments. Dashed lines indicate conjugated segments while colors denote rigid fragments. (a) Hexabenzocoronene: the π-conjugated system is both a rigid fragment and a conjugated segment. (b) Alq_3: the Al atom and each ligand are rigid fragments while the whole molecule is a conjugated segment. (c) Polythiophene: each repeat unit is a rigid fragment. A conjugated segment consists of one or more rigid fragments. One molecule can have several conjugated segments.

Self-assembling materials, such as soluble oligomers, discotic liquid crystals, block copolymers, partially crystalline polymers, etc., are the most complicated to study. The morphology of such systems often has several characteristic length scales and can be kinetically arrested in a thermodynamically non-equilibrium state. For such systems, the time- and length-scales of atomistic simulations might be insufficient to equilibrate or sample desired morphologies. In this case, systematic coarse-graining can be used to enhance sampling [6]. Note that the coarse-grained representation must reflect the structure of the atomistic system and allow for back-mapping to the atomistic resolution.

Here we assume that the topology and the coordinates of all atoms are known. VOTCA-CTP can read standard GROMACS topology files. Custom definitions of atomistic topology via XML files are also possible.

Conjugated segments and rigid fragments

With the morphology at hand, the next step is partitioning the system on hopping sites, or conjugated segments, and calculating charge transfer rates between them. Physically intuitive arguments can be used for the partitioning, which reflects the localization of the wave function of a charge. For most organic semiconductors, the molecular architecture includes relatively rigid, planar π-conjugated systems, which we will refer to as rigid fragments. A conjugated segment can contain one or more of such rigid fragments, which are linked by bonded degrees of freedom. The dynamics of these degrees of freedom evolves on timescales much slower than the frequency of the internal promoting mode. In some cases, e.g. glasses, it can be 'frozen' due to non-bonded interactions with the surrounding molecules.

To illustrate the concept of conjugated segments and rigid fragments, three representative molecular architectures are shown in figure 2. The first one is a typical discotic liquid crystal, hexabenzocoronene. It consists of a conjugated core to which side chains are attached to aid self-assembly and solution processing. In this case the orbitals localized on side chains do not participate in charge transport and the conjugated π-system is both, a rigid fragment and a conjugated segment. In Alq_3, a metal-coordinated compound, a charge carrier is delocalized over all three ligands. Hence, the whole molecule is one conjugated segment. Individual ligands are relatively rigid, while energies of the order of $k_\mathrm{B}T$ are sufficient to reorient them with respect to each other. Thus the Al atom and the three ligands are rigid fragments. In the case of a conjugated polymer, one molecule can consist of several conjugated segments, while each backbone repeat unit is a rigid fragment. Since the conjugation along the backbone can be broken due to large out-of-plane twists between two repeat units, an empirical criterion, based on the dihedral angle, can be used to partition the backbone on conjugated segments [8]. However, such intuitive partitioning is, to some extent, arbitrary and shall be validated by other methods [9–11].

After partitioning, an additional step is often required to remove bond length fluctuations introduced by molecular dynamics simulations, since they are already integrated out in the derivation of the rate expression. This is achieved by substituting respective molecular fragments with rigid, planar π-systems optimized using first-principles methods. Centers of mass and gyration tensors are used to align rigid fragments, though a custom definition of local axes is also possible. Such a procedure also minimizes discrepancies between the force-field and first-principles-based ground state geometries of conjugated segments, which might be important for calculations of electronic couplings, reorganization energies, and intramolecular driving forces.

Mapping file

The partitioning of the system on conjugated segments and rigid fragments is specified in the mapping file, map.xml. Using this input, ctp_map converts an atomistic configuration into a configuration with conjugated segments and rigid fragments and stores it in a state file:

Mapping using the GROMACS topology and trajectory

```
ctp_map -t topol.tpr -c traj.xtc -s map.xml -f state.sql
```

ctp_map reads in the GROMACS topology from topol.tpr, trajectory from traj.xtc, definitions of conjugated segments and rigid fragments from map.xml and outputs coordinates of segments and fragments to the state file, state.sql. After this step, dimensions of the simulation box, atom coordinates, etc, are stored in the state file.

Attention

VOTCA-CTP requires a wrapped GROMACS trajectory, i. e., all molecules should be whole in the snapshot.

Mapping file provides three different representations of the molecule: (i) partitioning on conjugated segments and rigid fragments (mdatoms), (ii) QM-optimized geometry (qmatoms), and (iii) partitioning on electrostatic multipoles (mpoles). An example of map.xml for a DCV2T molecule is shown in listing 1.

Listing 1: Examle of `map.xml` for DCV2T. Each rigid fragment (coarse-grained bead) is defined by a list of atoms. Atom numbers, names, and residue names should correspond to those used in GROMACS topology (see the corresponing listing 2 of the pdb file).

```
<topology> <!-- this file is used to conver an atomistic trajectory to conjugated segments -->
<molecules>
<molecule>
   <name>DCV2T-VOTCA</name> <!-- name of the conjugated molecule -->
   <mdname>DCV2T</mdname> <!-- name of the molecule in the MD topology, as in topol.top-->
   <segments>
   <segment>
        <name>DCV</name> <!-- name of the conjugated segment within the molecule -->
        <qmcoords>QC_FILES/DCV2T.xyz</qmcoords>  <!-- QM coordinates of the conjugated segment -->

        <!-- IZINDO INPUT -->
        <orbitals>QC_FILES/DCV2T.orb</orbitals><!-- INDO coefficients of MOs, Gaussian format -->
        <torbital_h>50</torbital_h><!-- HOMO orbital as in the log-file corresponding to DCV2T.orb) -->
        <torbital_e>50</torbital_e><!-- LUMO orbital as in the log-file corresponding to DCV2T.orb) -->

        <!-- XQMULTIPOLE INPUT -->
        <multipoles_n>MP_FILES/DCV2T.mps</multipoles_n><!-- Multipoles of a neutral molecule -->
        <multipoles_h>MP_FILES/DCV2T_h.mps</multipoles_h><!-- Multipoles of a cation -->
        <multipoles_e>MP_FILES/DCV2T_e.mps</multipoles_e><!-- Multipoles of an anion -->
        <map2md>0</map2md><!-- specifies if QM coordinates (0) or MD coordinates (1) are used for  partial charges. For MD the order and
              numbering in <mdatoms> and  <mpoles> must be identical-->

        <!-- EINTERNAL INPUT -->
        <U_cC_nN_h>0.0</U_cC_nN_h> <!-- Site energy      -->
        <U_nC_nN_h>0.1</U_nC_nN_h> <!-- Reorg. discharge -->
        <U_cN_cC_h>0.1</U_cN_cC_h> <!-- Reorg. charge    -->

        <!-- MD, QM, MP mappings -->
        <fragments>
        <fragment>
         <name>NI1</name> <!-- name of the rigid fragment within the segment -->
         <!-- list of atoms in the fragment resnum:resname:atomname -->
         <mdatoms>1:NIT:N1 1:NIT:CN1 1:NIT:N2  1:NIT:CN2 1:NIT:CC1 1:NIT:C1  1:NIT:HN1</mdatoms>
         <!-- corresponding ground state geometry atomnumber:atomtype  read from .xyz file-->
         <qmatoms>   20:N     19:C      14:N      13:C      12:C      11:C      23:H  </qmatoms>
         <!-- corresponding group state geometry multipoles read from .mps files -->
         <mpoles>    20:N     19:C      14:N      13:C      12:C      11:C      23:H  </mpoles>
         <!-- weights to determine the fragment center (here CoM is used) -->
         <weights>     14        12        14        12        12        12         1  </weights>
        <!-- three atoms define a local frame; two atoms for the rotationally invariant around some axis; one atom: fragment is assumed
              isotropic -->
         <localframe> 20 19 14 </localframe>
        <!-- Optional local frame for multipoles. If not provided, <localframe> is used. Useful when atom labels in the .mps and .xyz differ
              or more sites in the .mps file are present -->
         <localframe_mps> 20 19 14 </localframe_mps>
         <!-- Optional weights for multipoles. If not provided <weights> is used. Useful when atom labels in the .mps and .xyz differ or
               additional sites in the .mps file are present -->
        <weights_mps>     14        12        14        12        12        12          1  </weights_mps>
        <!-- Optional: specifies if an atom is virtual (1) or real (0) -->
        <virtual_mps>     0         0        0          0        0          0          0  </virtual_mps>
       </fragment>

      <fragment>
         <name>TH1</name>
         <mdatoms>2:THI:S1 2:THI:CA1 2:THI:CA2 2:THI:CB1 2:THI:CB2 2:THI:HC1 2:THI:HC2</mdatoms>
         <qmatoms>    7:S      8:C       6:C       9:C      10:C      24:H     25:H  </qmatoms>
         <mpoles>     7:S      8:C       6:C       9:C      10:C      24:H     25:H  </mpoles>
         <weights>     32        12        12        12        12         1         1  </weights>
         <localframe> 7 8 6 </localframe>
      </fragment>

      <fragment>
         <name>TH2</name>
         <mdatoms>3:THI:S1 3:THI:CA1 3:THI:CA2 3:THI:CB1 3:THI:CB2 3:THI:HC1 3:THI:HC2</mdatoms>
         <qmatoms>    3:S      4:C       2:C       5:C       1:C      26:H      27:H  </qmatoms>
         <weights>     32        12        12        12        12         1         1  </weights>
         <localframe> 3 4 2 </localframe>
      </fragment>

      <fragment>
         <name>NI2</name>
         <mdatoms>4:NIT:N1 4:NIT:CN1 4:NIT:N2  4:NIT:CN2 4:NIT:CC1 4:NIT:C1  4:NIT:HN1</mdatoms>
         <qmatoms>   22:N     21:C      18:N      17:C      16:C      15:C      28:H  </qmatoms>
         <mpoles>    22:N     21:C      18:N      17:C      16:C      15:C      28:H  </mpoles>
         <weights>     14        12        14       12        12        12         1  </weights>
         <localframe> 22 21 18 </localframe>
      </fragment>
      </fragments>
   </segment>
   </segments>
</molecule>
</molecules>
</topology>
```

The corresponding atom numbers, names, residue numbers, and residue types of DCV2T are shown in fig. 3 and in listing 2. DCV2T has two thiophene (THI) and two dicyanovinyl (NIT) residues, which are used to partition the molecule on four rigid fragments. Since the molecule is planar, it is represented by one conjugated segment.

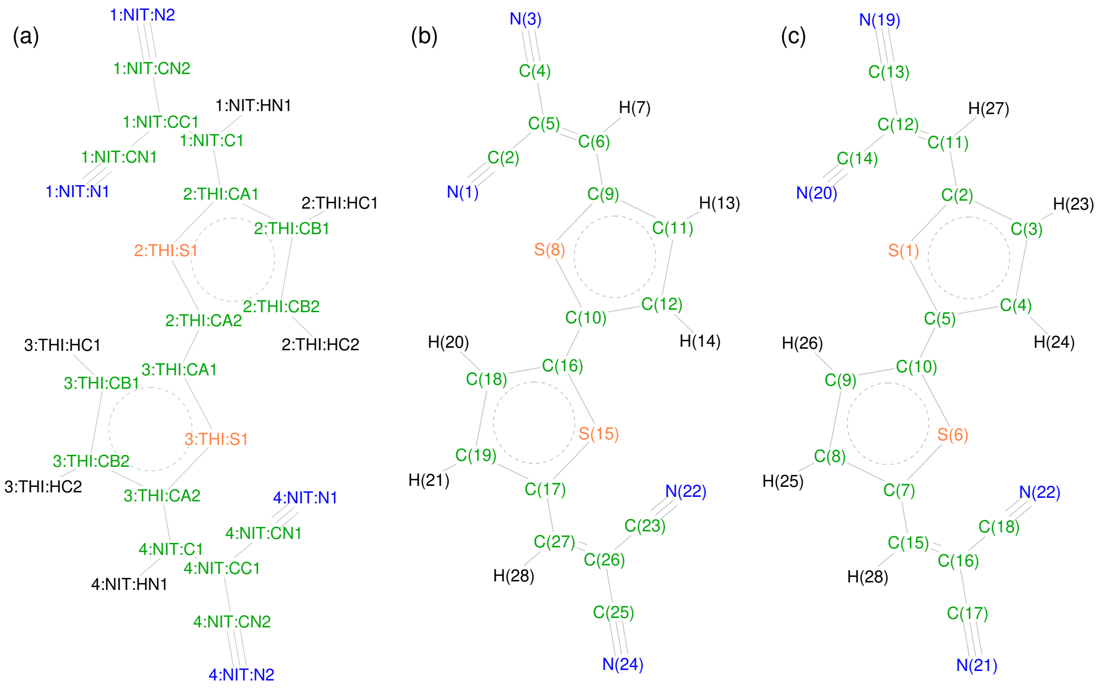


Figure 3: (a) DCV2T with atoms labelled according to `residue_number:residue_name:atom_name`. There are four residues and two residue types: thiophene (THI) and dicyanovinyl (NIT). The corresponding pdb file is shown in listing 2. Atom numbering is used to split conjugated segments on rigid fragments and to link atomistic (b) and quantum-mechanical (c) descriptions.

Listing 2: pdb file of DCV2T.

```
HETATM    1  N1  NIT     1       2.388   8.533  11.066  1.00  4.14           N
HETATM    2  CN1 NIT     1       1.984   9.553  10.718  1.00  2.54           C
HETATM    3  N2  NIT     1      -1.138  10.872  10.087  1.00  3.24           N
HETATM    4  CN2 NIT     1       0.003  10.871  10.213  1.00  2.37           C
HETATM    5  CC1 NIT     1       1.441  10.824  10.327  1.00  1.91           C
HETATM    6  C1  NIT     1       2.193  11.939  10.071  1.00  1.61           C
HETATM    7  HN1 NIT     1       1.715  12.710   9.872  1.00  1.97           H
HETATM    8  S1  THI     2       4.758  10.743  10.130  1.00  1.52           S
HETATM    9  CA1 THI     2       3.613  12.024   9.948  1.00  1.22           C
HETATM   10  CA2 THI     2       6.099  11.836   9.997  1.00  1.30           C
HETATM   11  CB1 THI     2       4.251  13.243   9.782  1.00  1.39           C
HETATM   12  CB2 THI     2       5.658  13.131   9.818  1.00  1.45           C
HETATM   13  HC1 THI     2       3.800  14.047   9.660  1.00  1.66           H
HETATM   14  HC2 THI     2       6.230  13.860   9.731  1.00  1.74           H
HETATM   15  S1  THI     3       8.803  12.414   9.882  1.00  1.38           S
HETATM   16  CA1 THI     3       7.456  11.347  10.094  1.00  1.37           C
HETATM   17  CA2 THI     3       9.940  11.122  10.152  1.00  1.42           C
HETATM   18  CB1 THI     3       7.873  10.048  10.355  1.00  1.73           C
HETATM   19  CB2 THI     3       9.267   9.926  10.399  1.00  1.82           C
HETATM   20  HC1 THI     3       7.288   9.335  10.487  1.00  2.05           H
HETATM   21  HC2 THI     3       9.704   9.123  10.576  1.00  2.21           H
HETATM   22  N1  NIT     4      11.235  14.572   9.094  1.00  3.08           N
HETATM   23  CN1 NIT     4      11.665  13.566   9.441  1.00  2.04           C
HETATM   24  N2  NIT     4      14.733  12.005  10.009  1.00  2.17           N
HETATM   25  CN2 NIT     4      13.590  12.149   9.933  1.00  1.77           C
HETATM   26  CC1 NIT     4      12.156  12.282   9.861  1.00  1.71           C
HETATM   27  C1  NIT     4      11.363  11.220  10.154  1.00  1.59           C
HETATM   28  HN1 NIT     4      11.813  10.440  10.389  1.00  1.89           H
```

Generation of the mapping file can be semi-automatized if the atom order, residue numbering and types in the atomistic morphology reflect partitioning on rigid fragments. This can be achieved already at the stage of preparing input files for atomistic simulations, by insuring that rigid fragments in the pdb files of all segments have different residue types and numbers. A DCV2T pdb file, shown in listing 2, is an example of such partitioning, where residue types (NIT and THI) and residue numbers (5-th column, from 1 to 4) uniquely define rigid fragments.

Such pdb files can be used to generate templates of GROMACS topologies, with the help of the pdb2top tool,

Creating a topol.top from a pdb file

```
ctp_tools -f state.sql -e pdb2top
```

and the mapping file, using the pdb2map tool,

Creating a map.xml from a pdb file

```
ctp_tools -f state.sql -e pdb2map
```

Both files need further refinement, but this procedure helps to avoid typos and inconsistencies between atomistic and coarse-grained morphologies.

Validating the mapping

In order to visually check the mapping one can use calculator trajectory2pdb of the ctp_dump program.

Writing a mapped trajectroy with ctp_dump

```
ctp_dump -f state.sql -e trajectory2pdb
```

ctp_dump reads in the state file created by ctp_map and outputs two trajectory files corresponding to the original and rigidified atom coordinates. To check the mapping, you can superimpose the three outputs: original atomistic, atomistic stored in the state file, and rigidified according to the ground state geometries. This can be done with, e.g., VMD. Note that ctp_dump can also output the list of segments,

List of conjugated segments

```
ctp_dump -f state.sql -e segments2xml
```

Alternatively, tdump of ctp_run reads in the state file and appends the coordinates to QM.pdb and MD.pdb. Make sure to delete old QM.pdb and MD.pdb if you want to create a new image.

Writing a mapped trajectory with tdump

```
ctp_run -f state.sql -o options.xml -e tdump
```

Custom topologies

GROMACS configurations and binary topologies (topology.tpr) can be used directly by the ctp_map program. In this case residue and atom names should be used to specify the coarse-grained topology and conjugated segments.

A custom topology can also be defined using an XML file. It is also possible to partially overwrite the information provided in, for example, GROMACS topology file.

Neighbor list

A list of neighboring conjugated segments, or neighbor list, contains all pairs of conjugated segments for which coupling elements, reorganization energies, site energy differences, and rates are evaluated.

Two segments are added to this list if the distance between centers of mass of any of their rigid fragments is below a certain cutoff. This allows neighbors to be selected on a criterion of minimum distance of approach rather than center of mass distance, which is useful for molecules with anisotropic shapes.

The neighbor list can be generated from the atomistic trajectory by using the `neighborlist` calculator. This calculator requires a cutoff, which can be specified in the `options.xml` file. The list is saved to the `state.sql` file:

Generating a neighbor list

```
ctp_run -o options.xml -f state.sql -e neighborlist
```

Sometimes it is convenient to filter the rigid fragments, for example if a conjugated segment has alkyl side chains and we do not want to add a pair of molecules which have conjugated cores far apart but side chains close to each other. In this case one can provide a list of "active" fragments. The search algorithm will only look at the pairs of segments, for which these fragments are located closer than the specified cutoff.

It can also happen that a completely custom neighbor list is required. An external list can be imported from a file which has the format pairID, segment1ID, segment2ID, segment1Type, segment2Type, e.g.,

```
1   1 2 DCV DCV
2   1 3 DCV DCV
3   2 3 DCV DCV
```

Reorganization energy

The reorganization energy λ_{ij} takes into account the change in nuclear (and dielectric) degrees of freedom as the charge moves from donor i to acceptor j. It has two contributions: intramolecular, $\lambda_{ij}^{\text{int}}$, which is due to reorganization of nuclear coordinates of the two molecules forming the charge transfer complex, and intermolecular (outersphere), $\lambda_{ij}^{\text{out}}$, which is due to the relaxation of the nuclear coordinates of the environment. In what follows we discuss how these contributions can be calculated.

Intramolecular

If intramolecular vibrational modes of the two molecules are treated classically, the rearrangement of their nuclear coordinates after charge transfer results in the dissipation of the internal reorganization energy, $\lambda_{ij}^{\text{int}}$. It can be computed from four points on the potential energy surfaces (PES) of both molecules in neutral and charged states, as indicated in figure 4.

Adding the contributions due to discharging of molecule i and charging of molecule j yields [12]

$$\lambda_{ij}^{\text{int}} = \lambda_i^{cn} + \lambda_j^{nc} = U_i^{nC} - U_i^{nN} + U_j^{cN} - U_j^{cC} \,. \tag{1}$$

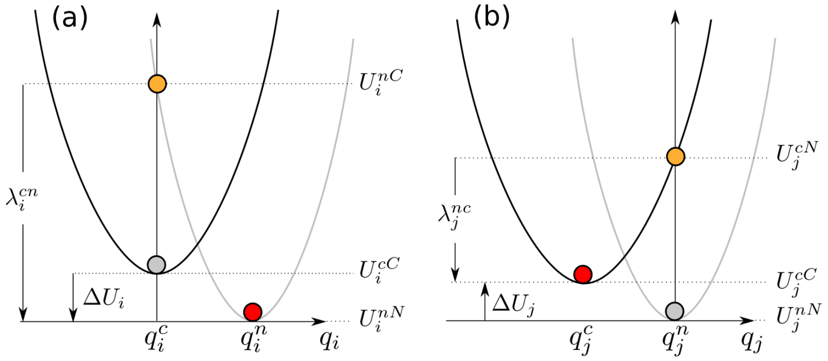


Figure 4: Potential energy surfaces of (a) donor and (b) acceptor in charged and neutral states. After the change of the charge state both molecules relax their nuclear coordinates. If all vibrational modes are treated classically, the total internal reorganization energy and the internal energy difference of the electron transfer reaction are $\lambda_{ij}^{\text{int}} = \lambda_i^{cn} + \lambda_j^{nc}$ and $\Delta E_{ij}^{\text{int}} = \Delta U_i - \Delta U_j$, respectively.

Here U_i^{nC} is the internal energy of the neutral molecule i in the geometry of its charged state (small n denotes the state and capital C the geometry). Similarly, U_j^{cN} is the energy of the charged molecule j in the geometry of its neutral state. Note that the PES of the donor and acceptor are not identical for chemically different compounds or for conformers of the same molecule. In this case $\lambda_i^{cn} \neq \lambda_j^{cn}$ and $\lambda_i^{nc} \neq \lambda_j^{nc}$. Thus $\lambda_{ij}^{\text{int}}$ is a property of the charge transfer complex, and not of a single molecule.

Intramolecular reorganization energies for discharging (λ^{cn}) and charging (λ^{nc}) of a molecule need to be determined using quantum-chemistry and given in `map.xml`. The values are written to the `state.sql` using the calculator `einternal` (see also internal energy):

Intramolecularl reorganization energies

```
ctp_run -o options.xml -f state.sql -e einternal
```

Outersphere

During the charge transfer reaction, also the molecules outside the charge transfer complex reorient and polarize in order to adjust for changes in electric potential, resulting in the outersphere contribution to the reorganization energy. $\lambda_{ij}^{\text{out}}$ is particularly important if charge transfer occurs in a polarizable environment. Assuming that charge transfer is much slower than electronic polarization but much faster than nuclear rearrangement of the environment, $\lambda_{ij}^{\text{out}}$ can be calculated from the electric displacement fields created by the charge transfer complex [13]

$$\lambda_{ij}^{\text{out}} = \frac{c_p}{2\epsilon_0} \int_{V^{\text{out}}} dV \left[\vec{D}_I(\vec{r}) - \vec{D}_F(\vec{r}) \right]^2 , \tag{2}$$

where ϵ_0 is the the permittivity of free space, $\vec{D}_{I,F}(\vec{r})$ are the electric displacement fields created by the charge transfer complex in the initial (charge on molecule i) and final (charge transferred to molecule j) states, V^{out} is the volume outside the complex, and $c_p = \frac{1}{\epsilon_{\text{opt}}} - \frac{1}{\epsilon_{\text{s}}}$ is the Pekar factor, which is determined by the low (ϵ_{s}) and high (ϵ_{opt}) frequency dielectric permittivities.

Eq. (2) can be simplified by assuming spherically symmetric charge distributions on molecules i and j with total charge e. Integration over the volume V^{out} outside of the two spheres of radii R_i and R_j centered on molecules i and j leads to the classical Marcus expression for the outersphere reorganization energy

$$\lambda_{ij}^{\text{out}} = \frac{c_p e^2}{4\pi\epsilon_0}\left(\frac{1}{2R_i} + \frac{1}{2R_j} - \frac{1}{r_{ij}}\right), \tag{3}$$

where r_{ij} is the molecular separation. While eq. (3) captures the main physics, e.g. predicts smaller outersphere reorganization energies (higher rates) for molecules at smaller separations, it often cannot provide quantitative estimates, since charge distributions are rarely spherically symmetric.

Alternatively, the displacement fields can be constructed using the atomic partial charges. The difference of the displacement fields at the position of an atom b_k outside the charge transfer complex (molecule $k \neq i, j$) can be expressed as

$$\vec{D}_I(\vec{r}_{b_k}) - \vec{D}_F(\vec{r}_{b_k}) = \sum_{a_i} \frac{q^c_{a_i} - q^n_{a_i}}{4\pi} \frac{(\vec{r}_{b_k} - \vec{r}_{a_i})}{|\vec{r}_{b_k} - \vec{r}_{a_i}|^3} + \sum_{a_j} \frac{q^n_{a_j} - q^c_{a_j}}{4\pi} \frac{(\vec{r}_{b_k} - \vec{r}_{a_j})}{|\vec{r}_{b_k} - \vec{r}_{a_j}|^3}, \tag{4}$$

where $q^n_{a_i}$ ($q^c_{a_i}$) is the partial charge of atom a of the neutral (charged) molecule i in vacuum. The partial charges of neutral and charged molecules are obtained by fitting their values to reproduce the electrostatic potential of a single molecule (charged or neutral) in vacuum. Assuming a uniform density of atoms, the integration in eq. (2) can be rewritten as a density-weighted sum over all atoms excluding those of the charge transfer complex.

The remaining unknown needed to calculate $\lambda_{ij}^{\text{out}}$ is the Pekar factor, c_p. In polar solvents $\epsilon_{\text{s}} \gg \epsilon_{\text{opt}} \sim 1$ and c_p is of the order of 1. In most organic semiconductors, however, molecular orientations are fixed and therefore the low frequency dielectric permittivity is of the same order of magnitude as ϵ_{opt}. Hence, c_p is small and its value is very sensitive to differences in the permittivities.

Outersphere reorganization energies for all pairs of molecules in the neighbor list can be computed from the atomistic trajectory by using the `eoutersphere` calculator.

Two methods can be used to compute $\lambda_{ij}^{\text{out}}$. The first method uses the atomistic partial charges of neutral and charged molecules from files specified in `map.xml` and eq. (2). The Pekar factor c_p and a cutoff radius based on molecular centers of mass have to be specified in the `options.xml` file.

If this method is computationally prohibitive, $\lambda_{ij}^{\text{out}}$ can be computed using eq. (3), which assumes spherical charge distributions on the molecules. In this case the radii of these spheres are specified in `segments.xml`, while the Pekar factor c_p is given in the `options.xml` file and no cutoff radius is needed.

The outer sphere reorganization energies are saved to the `state.sql` file:

Outersphere reorganization energy

```
ctp_run -o options.xml -f state.sql -e outersphere
```

Site energies

A charge transfer reaction between molecules i and j is driven by the site energy difference, $\Delta E_{ij} = E_i - E_j$. Since the transfer rate, ω_{ij}, depends exponentially on ΔE_{ij} (see eq. (31)) it is important to compute its distribution as accurately as possible. The total site energy difference has contributions due to externally applied electric field, electrostatic interactions, polarization effects, and internal energy differences. In what follows we discuss how to estimate these contributions by making use of first-principles calculations and polarizable force-fields.

External field

The contribution to the total site energy difference due to an external electric field $\vec{F}$ is given by $\Delta E_{ij}^{\text{ext}} = q\vec{F}\cdot\vec{r}_{ij}$, where $q = \pm e$ is the charge and $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$ is a vector connecting molecules i and j. For typical distances between small molecules, which are of the order of $1\,\mathrm{nm}$, and moderate fields of $F < 10^8\,\mathrm{V/m}$ this term is always smaller than $0.1\,\mathrm{eV}$.

Internal energy

The contribution to the site energy difference due to different internal energies (see figure 4) can be written as

$$\Delta E_{ij}^{\text{int}} = \Delta U_i - \Delta U_j = \left(U_i^{cC} - U_i^{nN}\right) - \left(U_j^{cC} - U_j^{nN}\right) , \tag{5}$$

where $U_i^{cC(nN)}$ is the total energy of molecule i in the charged (neutral) state and geometry. ΔU_i corresponds to the adiabatic ionization potential (or electron affinity) of molecule i, as shown in figure 4. For one-component systems and negligible conformational changes $\Delta E_{ij}^{\text{int}} = 0$, while it is significant for donor-acceptor systems.

Internal energies determined using quantum-chemistry need to be specified in `map.xml`. The values are written to the `state.sql` using the calculator `einternal` (see also intramolecular reorganization energy):

Internal energies

```
ctp_run  -o options.xml  -f state.sql  -e einternal
```

Electrostatic interactions

We represent the molecular charge density by choosing multiple expansion sites ("polar sites") per molecule in such a way as to accurately reproduce the molecular electrostatic potential (ESP), with a set of suitably chosen multipole moments $\{Q^a_{lk}\}$ (in spherical-tensor notation) allocated to each site. The expression for the electrostatic interaction energy between two molecules A and B in the multi-point expansion includes an implicit sum over expansion sites $a\epsilon A$ and $b\epsilon B$,

$$U_{AB} = \sum_{a\epsilon A}\sum_{b\epsilon B} \hat{Q}^a_{l_1k_1} T^{a,b}_{l_1k_1l_2k_2} \hat{Q}^b_{l_2k_2} \equiv \hat{Q}^a_{l_1k_1} T^{a,b}_{l_1k_1l_2k_2} \hat{Q}^b_{l_2k_2}, \tag{6}$$

where we have used the Einstein sum convention for the site indices a and b on the right-hand side of the equation, in addition to the sum convention that is in place for the multipole-moment components $t \equiv l_1k_1$ and $u \equiv l_2k_2$. The $T^{a,b}_{l_1k_1l_2k_2}$ are tensors that mediate the interaction between a multipole component l_1k_1 on site a with the moment l_2k_2 on site b. If we include the molecular environment into a perturbative term W to enter in the single-molecule Hamiltonian, the above expression is exactly the first-order correction to the energy where the quantum-mechanical detail has been absorbed in classical multipole moments.

The are a number of strategies how to arrive at such a collection of *distributed multipoles*. They can be classified according to whether the multipoles are derived (a) from the electrostatic potential generated by the SCF charge density or (b) from a decomposition of the wavefunction itself. Here, we will only draft two of those approaches, CHELPG [14] from category (a) and DMA [15] from category (b).

The CHELPG (CHarges from ELectrostatic Potentials, Grid-based) method relies on performing a least-squares fit of atom-placed charges to reproduce the electrostatic potential as evaluated from the SCF density on a regularly spaced grid [14]. The fitted charges result from minimizing the Lagrangian function [16]

$$z(\{q_i\}) = \sum_{k=1}^{M}\left(\phi(\vec{r}_k) - \sum_{i=1}^{N}\frac{1}{4\pi\varepsilon_0}\frac{q_i}{|\vec{r}_i - \vec{r}_k|}\right) + \lambda\left(q_{\text{mol}} - \sum_{i=1}^{N} q_i\right), \tag{7}$$

with M grid points, N atomic sites, the set of atomic partial charges $\{q_i\}$ and the SCF potential ϕ. The Lagrange multiplier λ constrains the sum of the fitted charges to the molecular charge q_{mol}. The main difference from other fitting schemes [17] is the algorithm that selects the positions at which the potential is evaluated (we note that the choice of grid points can have substantial effects especially for bulky molecules). Clearly, the CHELPG method can be (and has been) extended to include higher atomic multipoles. It should be noted, however, how already the inclusion of atomic dipoles hardly improves the parametrization, and can in fact be harmful to its conformational stability.

The Distributed-Multipole-Analysis (DMA) approach [15, 18], developed by A. Stone, operates directly on the quantum-mechanical density matrix, expanded in terms of atom- and bond-centered Gaussian functions $\chi_\alpha = R_{LK}(\vec{x} - \vec{s}_\alpha)\exp[-\zeta(\vec{x} - \vec{s}_\alpha)^2]$,

$$\rho(\vec{x}) = \sum_{\alpha,\beta} \rho_{\alpha\beta}\chi_\alpha(\vec{x} - \vec{s}_\alpha)\chi_\beta(\vec{x} - \vec{s}_\beta). \tag{8}$$

The aim is to compute multipole moments according in a distributed fashion: If we use that the overlap product $\chi_\alpha\chi_\beta$ of two Gaussian basis functions yields itself a Gaussian centered at $\vec{P} = (\zeta_\alpha\vec{s}_\alpha + \zeta_\beta\vec{s}_\beta)/(\zeta_\alpha + \zeta_\beta)$, it is possible to proceed in two steps: First, we compute the multipole moments associated with a specific summand in the density matrix, referred to the overlap center $\vec{P}$:

$$Q_{LK}[\vec{P}] = -\int R_{LK}(\vec{x} - \vec{P})\rho_{\alpha\beta}\chi_\alpha\chi_\beta d^3x. \tag{9}$$

Second, we transfer the resulting $Q_{lk}[\vec{P}]$ to the position $\vec{S}$ of a polar site according to the rule [15]

$$Q_{nm}[\vec{S}] = \sum_{l=0}^{L}\sum_{k=-l}^{l}\left[\begin{pmatrix} n+m \\ l+k \end{pmatrix}\begin{pmatrix} n-m \\ l-k \end{pmatrix}\right]^{1/2} R_{n-l,m-k}(\vec{S} - \vec{P}) \cdot Q_{lk}[\vec{P}]. \tag{10}$$

Note how this requires a rule for the choice of the expansion site to which the multipole moment should be transferred. In the near past [18], the nearest-site algorithm, which allocates the multipole moments to the site closest to the overlap center, was replaced for diffuse functions by an algorithm based on a smooth weighting function in conjunction with grid-based integration methods in order to decrease the basis-set dependence of the resulting set of distributed multipoles.

One important advantage of the DMA approach over fitting algorithms such as CHELPG or Merz-Kollman (MK) is that higher-order moments can also be derived without too large an ambiguity.

The 'mps' file format used by VOTCA for the definition of distributed multipoles (as well as point polarizabilities, see subsequent section) is based on the GDMA punch format of A. Stone's GDMA program [18] (the punch output file can be immediately plugged into VOTCA without any conversions to be applied). Furthermore the log-file of different QM packages (currently `Gaussian`, `Turbomole` and `NWChem`) may be fed into the `log2mps` tool, which will subsequently generate the appropriate mps-file.

Read in ESP charges from a QM log file

```
ctp_tools -o options.xml -e log2mps
```

Induction interactions

If we in addition to the permanent set of multipole moments $\{Q_t^a\}$ allow for induced moments $\{\Delta Q_t^a\}$ and penalize their generation with a bilinear form (giving rise to a strictly positive contribution to the energy),

$$U_{\text{int}} = \frac{1}{2}\sum_{A} \Delta Q_t^a \eta_{tt'}^{aa'} \Delta Q_{t'}^{a'}, \tag{11}$$

it can be shown that the induction contribution to the site energy evaluates to an expression where all interactions between induced moments have cancelled out, and interactions between permanent and induced moments are scaled down by $1/2$ [19]:

$$U_{pu} = \frac{1}{2}\sum_{A}\sum_{B>A}\left[\Delta Q_t^a T_{tu}^{ab} Q_u^b + \Delta Q_t^b T_{tu}^{ab} Q_u^a\right]. \tag{12}$$

This term can be viewed as the second-order (induction) correction to the molecular interaction energy. The sets of $\{Q_t^a\}$ are solved for self-consistently via

$$\Delta Q_t^a = -\sum_{B\neq A} \alpha_{tt'}^{aa'} T_{t'u}^{a'b}(Q_u^b + \Delta Q_u^b), \tag{13}$$

where the polarizability tensors $\alpha_{tt'}^{aa'}$ are given by the inverse of $\eta_{tt'}^{aa'}$.

With eqs. 13 and 12 we have at hand expressions that allow us to compute the induction energy contribution to site energies in an iterative manner based on a set of molecular distributed multipoles $\{Q_t^a\}$ and polarizabilities $\{\alpha_{tt'}^{aa'}\}$. We have drafted in the previous section how to obtain the former from a wavefunction decomposition or fitting scheme (GDMA, CHELPG). The $\{\alpha_{tt'}^{aa'}\}$ can be derived formally (or rather: read off) from a perturbative expansion of the molecular interaction. In this work we make use of the Thole model [20, 21] as a semi-empirical approach to obtain the sought-after point polarizabilities in the local dipole approximation, that is, $[\alpha_{tt'}^{aa'}] = \alpha_{tt'}^{aa'}\delta_{t\beta}\delta_{t'\beta}\delta_{aa'}$, where $\beta\epsilon\{x, y, z\}$ references the dipole-moment component.

The Thole model is based on a modified dipole-dipole interaction, which can be reformulated in terms of the interaction of smeared charge densities. This has been shown to be necessary due to the divergent head-to-tail dipole-dipole interaction that otherwise results at small interseparations on the Å scale [20–22]. Smearing out the charge distribution mimics the nature of the QM wavefunction, which effectively guards against this unphysical polarization catastrophe. Since the point dipoles however only react individually to the external field, any correlation effects as were still accounted for in the $\{\alpha_{tt'}^{aa'}\}$ are lost, except perhaps those correlations that are due to the mere classical field interaction.

The smearing of the nuclei-centered multipole moments is obtained via a fractional charge density $\rho_f(\vec{u})$ which should be normalized to unity and fall off rapidly as of a certain radius $\vec{u} = \vec{u}(\vec{R})$. The latter is related to the physical distance vector $\vec{R}$ connecting two interacting sites via a linear scaling factor that takes into account the magnitude of the isotropic site polarizabilities α^a. This isotropic fractional charge density gives rise to a modified potential

$$\phi(u) = -\frac{1}{4\pi\varepsilon_0}\int_0^u 4\pi u'\rho(u')du' \tag{14}$$

We can relate the multipole interaction tensor $T_{ij\ldots}$ (this time in Cartesian coordinates) to the fractional charge density in two steps: First, we rewrite the tensor in terms of the scaled distance vector $\vec{u}$,

$$T_{ij\ldots}(\vec{R}) = f(\alpha^a\alpha^b)\, t_{ij\ldots}(\vec{u}(\vec{R}, \alpha^a\alpha^b)), \tag{15}$$

where the specific form of $f(\alpha^a\alpha^b)$ results from the choice of $u(\vec{R}, \alpha^a\alpha^b)$. Second, we demand that the smeared interaction tensor $t_{ij\ldots}$ is given as usual by the appropriate derivative of the potential in eq. 14,

$$t_{ij\ldots}(\vec{u}) = -\partial_{u_i}\partial_{u_j}\ldots\phi(\vec{u}). \tag{16}$$

It turns out that for a suitable choice of $\rho_f(\vec{u})$, the modified interaction tensors can be rewritten in such a way that powers n of the distance $R = |\vec{R}|$ are damped with a damping function $\lambda_n(\vec{u}(\vec{R}))$ [23].

There is a large number of fractional charge densities $\rho_f(\vec{u})$ that have been tested for the purpose of giving best results for the molecular polarizability as well as interaction energies. Note how a great advantage of the Thole model is the exceptional transferability of the atomic polarizabilities to compounds not used for the fitting procedure [21]. In fact, for most organic molecules, a fixed set of atomic polarizabilities ($\alpha_C = 1.334$, $\alpha_H = 0.496$, $\alpha_N = 1.073$, $\alpha_O = 0.873$, $\alpha_S = 2.926$ Å^3) based on atomic elements yields satisfactory results.

VOTCA implements the Thole model with an exponentially-decaying fractional charge density

$$\rho(u) = \frac{3a}{4\pi}\exp(-au^3), \tag{17}$$

where $\vec{u}(\vec{R}, \alpha^a\alpha^b) = \vec{R}/(\alpha^a\alpha^b)^{1/6}$ and the smearing exponent $a = 0.39$ (which can however be changed from the program options), as used in the AMOEBA force field [23].

Even though the Thole model performs very well for many organic compounds with only the above small set of element-based polarizabilities, conjugated molecules may require a more intricate parametrization. The simplest approach is to resort to scaled polarizabilities to match the effective molecular polarizable volume $V \sim \alpha_x\alpha_y\alpha_z$ as predicted by QM calculations (here $\alpha_x, \alpha_y, \alpha_z$ are the eigenvalues of the molecular polarizability tensor). The `molpol` tool assists with this task, it self-consistently calculates the Thole polarizability for an input mps-file and optimizes (if desired) the atomic polarizabilities in the above simple manner.

Generate Thole-type polarizabilites for a segment

```
ctp_tools  -o options.xml  -e molpol
```

The electrostatic and induction contribution to the site energy is evaluated by the `emultipole` calculator. Atomistic partial charges for charged and neutral molecules are taken from mps-files (extended GDMA format) specified in `map.xml`. Note that, in order to speed up calculations for both methods, a cut-off radius (for the molecular centers of mass) can be given in `options.xml`. Threaded execution is advised.

Electrostatic and induction corrections

```
ctp_run  -o options.xml  -f state.sql  -e emultipole
```

Furthermore available are `zmultipole`, which extends `emultipole` to allow for an electrostatic buffer layer (loosely related to the z-buffer in OpenGL, hence the name) and anisotropic point polarizabilities. For the interaction energy of charged clusters of any user-defined composition (Frenkel states, CT states, ...), `xqmultipole` can be used.

Interaction energy of charged molecular clusters embedded in a molecular environment

```
ctp_parallel  -o options.xml  -f state.sql  -e xqmultipole
```

Ewald sums

This section is a practical guide for doing electrostatic calculations in slabs using aperiodic Ewald scheme described in [1].

First, you will need to generate the required quantum mechanical reference, comprising molecular charge density and polarizability for all charge states (neutral, cationic, anionic). For example, you can use GAUSSIAN to do this:

```
...
#p b3lyp/6-31+g(d,p) pop(full,chelpg) polar(dipole) nosymm test
...
```

Afterwords, you have to generate all required *mps*-files with distributed multipoles and polarizabilities. The options file should point to the QM *log*-files, as well as contain the target molecular polarizability tensor in upper-diagonal order xx xy xz yy yz zz in units of Å^3.

```
$ ctp_tools -e log2mps -o options.xml
$ ctp_tools -e molpol -o options.xml

<options>
    <log2mps>
        <package>gaussian</package>
        <logfile>input.log</logfile>
        <mpsfile></mpsfile>
    </log2mps>
    <molpol>
        <mpsfiles>
            <input>input.mps</input>
            <output>output.mps</output>
            <polar>output.xml</polar>
        </mpsfiles>
        <induction>
            <expdamp>0.39000</expdamp>
            <wSOR>0.30000</wSOR>
            <maxiter>1024</maxiter>
            <tolerance>0.00001</tolerance>
        </induction>
        <target>
            <optimize>true</optimize>
            <molpol>77 0 0 77 0 77</molpol>
            <tolerance>0.00001</tolerance>
        </target>
    </molpol>
</options>
```

Next, generate the *mps* table, which relates the state of the molecule (neutral, cation, anion) with a corresponding electrostatic representation. This is provided by the *stateserver*. The resulting output file has the default name “mps.tab”:

```
$ ctp_run -e stateserver -o options.xml -f state.sql

<options>
    <stateserver>
        <keys>mps</keys>
    </stateserver>
</options>
```

Next, generate the job file. This job file lists the electrostatic configurations that are to be investigated. It can either be composed by hand or generated automatically from the *sql* file via the *jobwriter*, which at present takes either “mps.chrg”, “mps.single” or “mps.ct” as key, where the latter resorts to a neighbor list. The resulting *xml*-file has the default name “jobwriter.mps.chrg.xml”.

```
$ ctp_run -e jobwriter -o options.xml -f state.sql

<options>
```

```
    <jobwriter>
        <keys>mps.chrg</keys>
 <single_id>10</single_id>
 <cutoff>3</cutoff>
    </jobwriter>
</options>
```

The input string in the job file for the long-range corrected calculators has the same format as for the *xqmultipole* calculator, “id1:name1:mps1 id2:name2:mps2 ...”. See sample below.

```
 <jobs>
    <job>
        <id>1</id>
        <tag>1e:2h</tag>
        <input>1:spl:MP_FILES/spl_e.mps 2:spl:MP_FILES/spl_h.mps</input>
        <status>AVAILABLE</status>
    </job>
    <job>
        ...
    </job>
    ...
</jobs>
```

Next, generate the *ptop*-file that stores the induction state of the neutral *background*. The responsible *ewdbgpol* calculator can be run in a threaded fashion, depending on system size. The resulting *ptop*-file has the default name “bgp_main.ptop”.

```
$ ctp_run -e ewdbgpol -o options.xml -f state.sql -t 8

<options>
    <ewdbgpol>
        <multipoles>system.xml</multipoles>
        <control>
            <mps_table>mps.tab</mps_table>
            <pdb_check>1</pdb_check>
        </control>
        <coulombmethod>
            <method>ewald</method>
            <cutoff>6</cutoff>
            <shape>xyslab</shape>
        </coulombmethod>
        <polarmethod>
            <method>thole</method>
            <wSOR_N>0.350</wSOR_N>
            <aDamp>0.390</aDamp>
        </polarmethod>
        <convergence>
            <energy>1e-05</energy>
            <kfactor>100</kfactor>
            <rfactor>6</rfactor>
        </convergence>
    </ewdbgpol>
</options>
```

Finally, run the energy computation using *pewald3d*. This job calculator is wrapped by the ctp_parallel executable, which allows for communication between different processes via the job and state file. Unfortunately, communication, though guarded by a file lock, may fail on some architectures in the event of frequent accesses to the job file. This frequency can be controlled by the -c/–cache argument, which defines the number of jobs that are loaded in *one* batch by a specific process/node.

```
$ ctp_parallel -e pewald3d -o options.xml -f /absolute/path/to/state.sql -s 0 -t 8 -c
```

```
<options>
    <ewald>
        <jobcontrol>
            <job_file>/absolute/path/to/jobs.xml</job_file>
        </jobcontrol>
        <multipoles>
            <mapping>system.xml</mapping>
            <mps_table>mps.tab</mps_table>
            <polar_bg>bgp_main.ptop</polar_bg>
            <pdb_check>0</pdb_check>
        </multipoles>
        <coulombmethod>
            <method>ewald</method>
            <cutoff>8</cutoff>
            <shape>xyslab</shape>
            <save_nblist>false</save_nblist>
        </coulombmethod>
        <polarmethod>
            <method>thole</method>
            <induce>1</induce>
            <cutoff>4</cutoff>
            <tolerance>0.001</tolerance>
            <radial_dielectric>4.0</radial_dielectric>
        </polarmethod>
        <tasks>
            <calculate_fields>true</calculate_fields>
            <polarize_fg>true</polarize_fg>
            <evaluate_energy>true</evaluate_energy>
            <apply_radial>false</apply_radial>
        </tasks>
        <coarsegrain>
            <cg_background>false</cg_background>
            <cg_foreground>false</cg_foreground>
            <cg_radius>3</cg_radius>
            <cg_anisotropic>true</cg_anisotropic>
        </coarsegrain>
        <convergence>
            <energy>1e-05</energy>
            <kfactor>100</kfactor>
            <rfactor>6</rfactor>
        </convergence>
    </ewald>
</options>
```

Parse the output. The results from the computation are stored in the same job file that was supplied to the calculator. The key data is provided in the *output/summary* section and consists of the electrostatic and induction contributions *output/summary/eindu* and *output/summary/estat*. Note that only configuration energy *differences* carry meaning. The parsing is best done by script, as the “-j/–jobs read” option for pewald3d is not yet implemented.

```
 <jobs>
    <job>
        <id>1</id>
        <tag>1e:2h</tag>
        <input>1:spl:MP_FILES/spl_e.mps 2:spl:MP_FILES/spl_h.mps</input>
        <status>COMPLETE</status>
        <host>thop76:5476</host>
        <time>17:22:56</time>
        <output>
            <summary>
                <type>neutral</type>
                <xyz unit="nm">-0.1750000 -0.1750000 -5.4250000</xyz>
                <total unit="eV">-3.2112834</total>
                <estat unit="eV">-2.3753255</estat>
                <eindu unit="eV">-0.8359579</eindu>
            </summary>
            <terms_i>
                <F-00-01-11>-3.32999e+00 -3.32481e-01 +4.06829e-02</F-00-01-11>
                <M-00-11--->+1.33689e-01 +4.74490e-01</M-00-11--->
                <E-PP-PU-UU>-2.37533e+00 -7.37812e-01 -2.73212e-18</E-PP-PU-UU>
            </terms_i>
            <terms_o>
                <R-pp-pu-uu>-1.89357e-01 = +2.69583e-08 ...</R-pp-pu-uu>
                <K-pp-pu-uu>-7.31703e-03 = -2.69583e-08 ...</K-pp-pu-uu>
                <O-pp-pu-uu>+0.00000e+00 = +0.00000e+00 ...</O-pp-pu-uu>
                <J-pp-pu-uu>+5.14048e-04 = -2.99187e-17 ...</J-pp-pu-uu>
                <C-pp-pu-uu>+1.51186e-03 = +0.00000e+00 ...</C-pp-pu-uu>
                <Q-pp-pu-uu>+0.00000e+00 = +0.00000e+00 ...</Q-pp-pu-uu>
            </terms_o>
            <shells>
                <FGC>1874</FGC>
                <FGN>1874</FGN>
                <MGN>54429</MGN>
                <BGN>36</BGN>
                <BGP>52</BGP>
                <QM0>2</QM0>
                <MM1>1872</MM1>
                <MM2>0</MM2>
            </shells>
            <timing>
                <t_total unit="min">5.29</t_total>
                <t_wload unit="min">0.00 2.24 0.88 2.18</t_wload>
            </timing>
        </output>
    </job>
    <job>
     ...
    </job>
  ...
</jobs>
```

Electronic couplings

The electronic transfer integral element J_{ij} entering the Marcus rates in eq. (31) is defined as

$$J_{ij} = \left\langle \phi_i \middle| \hat{H} \middle| \phi_j \right\rangle , \tag{18}$$

where ϕ_i and ϕ_j are diabatic wavefunctions, localized on molecule i and j respectively, participating in the charge transfer, and $\hat{H}$ is the Hamiltonian of the formed dimer. Within the frozen-core approximation, the usual choice for the diabatic wavefunctions ϕ_i is the highest occupied molecular orbital (HOMO) in case of hole transport, and the lowest unoccupied molecular orbital (LUMO) in the case of electron transfer, while $\hat{H}$ is an effective single particle Hamiltonian, e.g. Fock or Kohn-Sham operator of the dimer. As such, J_{ij} is a measure of the strength of the electronic coupling of the frontier orbitals of monomers mediated by the dimer interactions.

Intrinsically, the transfer integral is very sensitive to the molecular arrangement, i.e. the distance and the mutual orientation of the molecules participating in charge transport. Since this arrangement can also be significantly influenced by static and/or dynamic disorder [24–28], it is essential to calculate J_{ij} explicitly for each hopping pair within a realistic morphology. Considering that the number of dimers for which eq. (18) has to be evaluated is proportional to the number of molecules times their coordination number, computationally efficient and at the same time quantitatively reliable schemes are required.

Projection

An approach for the determination of the transfer integral that can be used for any single-particle electronic structure method (Hartree-Fock, DFT, or semiempirical methods) is based on the projection of monomer orbitals on a manifold of explicitly calculated dimer orbitals. This dimer projection (DIPRO) technique including an assessment of computational parameters such as the basis set, exchange-correlation functionals, and convergence criteria is presented in detail in ref. [5]. A brief summary of the concept is given below.

We start from an effective Hamiltonian [1]

$$\hat{H}^{\text{eff}} = \sum_i \epsilon_i \hat{a}_i^\dagger \hat{a}_i + \sum_{j \neq i} J_{ij} \hat{a}_i^\dagger \hat{a}_j + c.c. \tag{19}$$

where $\hat{a}_i^\dagger$ and $\hat{a}_i$ are the creation and annihilation operators for a charge carrier located at the molecular site i. The electron site energy is given by ϵ_i, while J_{ij} is the transfer integral between two sites i and j. We label their frontier orbitals (HOMO for hole transfer, LUMO for electron transfer) ϕ_i and ϕ_j, respectively. Assuming that the frontier orbitals of a dimer (adiabatic energy surfaces) result exclusively from the interaction of the frontier orbitals of monomers, and consequently expand them in terms of ϕ_i and ϕ_j. The expansion coefficients, $\bar{\mathbf{C}}$, can be determined by solving the secular equation

$$(\mathbf{H} - E\mathbf{S})\bar{\mathbf{C}} = 0 \tag{20}$$

where $\mathbf{H}$ and $\mathbf{S}$ are the Hamiltonian and overlap matrices of the system, respectively. These matrices can be written explicitly as

$$\mathbf{H} = \begin{pmatrix} e_i & H_{ij} \\ H_{ij}^* & e_j \end{pmatrix} \qquad \mathbf{S} = \begin{pmatrix} 1 & S_{ij} \\ S_{ij}^* & 1 \end{pmatrix} \tag{21}$$

[1] we use following notations: a - number, $\bar{\mathbf{a}}$ - vector, $\mathbf{A}$ - matrix, $\hat{A}$ - operator

with

$$\begin{aligned} e_i &= \langle\phi_i|\,\hat{H}\,|\phi_i\rangle & \qquad H_{ij} &= \langle\phi_i|\,\hat{H}\,|\phi_j\rangle \\ e_j &= \langle\phi_j|\,\hat{H}\,|\phi_j\rangle & \qquad S_{ij} &= \langle\phi_j|\,\phi_j\rangle \end{aligned} \tag{22}$$

The matrix elements $e_{i(j)}$, H_{ij}, and S_{ij} entering eq. (21) can be calculated via projections on the dimer orbitals (eigenfunctions of $\hat{H}$) $\{|\phi_n^{\mathrm{D}}\rangle\}$ by inserting $\hat{1} = \sum_n |\phi_n^{\mathrm{D}}\rangle\langle\phi_n^{\mathrm{D}}|$ twice. We exemplify this explicitly for H_{ij} in the following

$$H_{ij} = \sum_{nm} \left\langle \phi_i \,\middle|\, \phi_n^{\mathrm{D}} \right\rangle \langle\phi_n^{D}|\,\hat{H}\left|\phi_m^{\mathrm{D}}\right\rangle \left\langle \phi_m^{\mathrm{D}} \,\middle|\, \phi_j \right\rangle. \tag{23}$$

The Hamiltonian is diagonal in its eigenfunctions, $\langle\phi_n^{\mathrm{D}}|\,\hat{H}\,|\phi_m^{\mathrm{D}}\rangle = E_n\delta_{nm}$. Collecting the projections of the frontier orbitals $|\phi_{i(j)}\rangle$ on the n-th dimer state $\left(\bar{\mathbf{V}}_{(i)}\right)_n = \langle\phi_i\,|\,\phi_n^{\mathrm{D}}\rangle$ and $\left(\bar{\mathbf{V}}_{(j)}\right)_n = \langle\phi_j\,|\,\phi_n^{\mathrm{D}}\rangle$ respectively, into vectors we obtain

$$H_{ij} = \bar{\mathbf{V}}_{(i)}\mathbf{E}\bar{\mathbf{V}}_{(j)}^{\dagger}. \tag{24}$$

What is left to do is determine these projections $\bar{\mathbf{V}}_{(k)}$. In all practical calculations the molecular orbitals are expanded in basis sets of either plane waves or of localized atomic orbitals $|\varphi_\alpha\rangle$. We will first consider the case that the calculations for the monomers are performed using a counterpoise basis set that is commonly used to deal with the basis set superposition error (BSSE). The basis set of atom-centered orbitals of a monomer is extended to the one of the dimer by adding the respective atomic orbitals at virtual coordinates of the second monomer. We can then write the respective expansions as

$$|\phi_k\rangle = \sum_\alpha \lambda_\alpha^{(k)}\,|\varphi_\alpha\rangle \qquad \text{and} \qquad \left|\phi_n^{\mathrm{D}}\right\rangle = \sum_\alpha D_\alpha^{(n)}\,|\varphi_\alpha\rangle \tag{25}$$

where $k = i, j$. The projections can then be determined within this common basis set as

$$\left(\bar{\mathbf{V}}_k\right)_n = \left\langle \phi_k \,\middle|\, \phi_n^{\mathrm{D}} \right\rangle = \sum_\alpha \lambda_\alpha^{(k)}\,\langle\alpha|\sum_\beta D_\beta^{(n)}\,|\beta\rangle = \bar{\boldsymbol{\lambda}}_{(k)}^{\dagger}\mathcal{S}\bar{\mathbf{D}}_{(n)} \tag{26}$$

where $\mathcal{S}$ is the overlap matrix of the atomic basis functions. This allows us to finally write the elements of the Hamiltonian and overlap matrices in eq. (21) as:

$$\begin{aligned} H_{ij} &= \bar{\boldsymbol{\lambda}}_{(i)}^{\dagger}\mathcal{S}\mathbf{D}\mathbf{E}\mathbf{D}^{\dagger}\mathcal{S}^{\dagger}\bar{\boldsymbol{\lambda}}_{(j)} \\ S_{ij} &= \bar{\boldsymbol{\lambda}}_{(i)}^{\dagger}\mathcal{S}\mathbf{D}\mathbf{D}^{\dagger}\mathcal{S}^{\dagger}\bar{\boldsymbol{\lambda}}_{(j)} \end{aligned} \tag{27}$$

Since the two monomer frontier orbitals that form the basis of this expansion are not orthogonal in general ($\mathbf{S} \neq \mathbf{1}$), it is necessary to transform eq. (20) into a standard eigenvalue problem of the form

$$\mathbf{H}^{\mathrm{eff}}\bar{\mathbf{C}}^{\mathrm{eff}} = E\bar{\mathbf{C}}^{\mathrm{eff}} \tag{28}$$

to make it correspond to eq. (19). According to Löwdin such a transformation can be achieved by

$$\mathbf{H}^{\mathrm{eff}} = \mathbf{S}^{-1/2}\mathbf{H}\mathbf{S}^{-1/2}. \tag{29}$$

This then yields an effective Hamiltonian matrix in an orthogonal basis, and its entries can directly be identified with the site energies ϵ_i and transfer integrals J_{ij}:

$$\mathbf{H}^{\mathrm{eff}} = \begin{pmatrix} e_i^{\mathrm{eff}} & H_{ij}^{\mathrm{eff}} \\ H_{ij}^{*,\mathrm{eff}} & e_j^{\mathrm{eff}} \end{pmatrix} = \begin{pmatrix} \epsilon_i & J_{ij} \\ J_{ij}^{*} & \epsilon_j \end{pmatrix} \tag{30}$$

The calculation of one electronic coupling element requires the overlap matrix of atomic orbitals $\mathcal{S}$, the expansion coefficients for monomer $\bar{\boldsymbol{\lambda}}_{(k)} = \{\lambda_\alpha^{(k)}\}$ and dimer orbitals $\bar{\mathbf{D}}_{(n)} = \{D_\alpha^{(n)}\}$, as well as the orbital energies E_n of the dimer are required as input. In practical situations, performing self-consistent

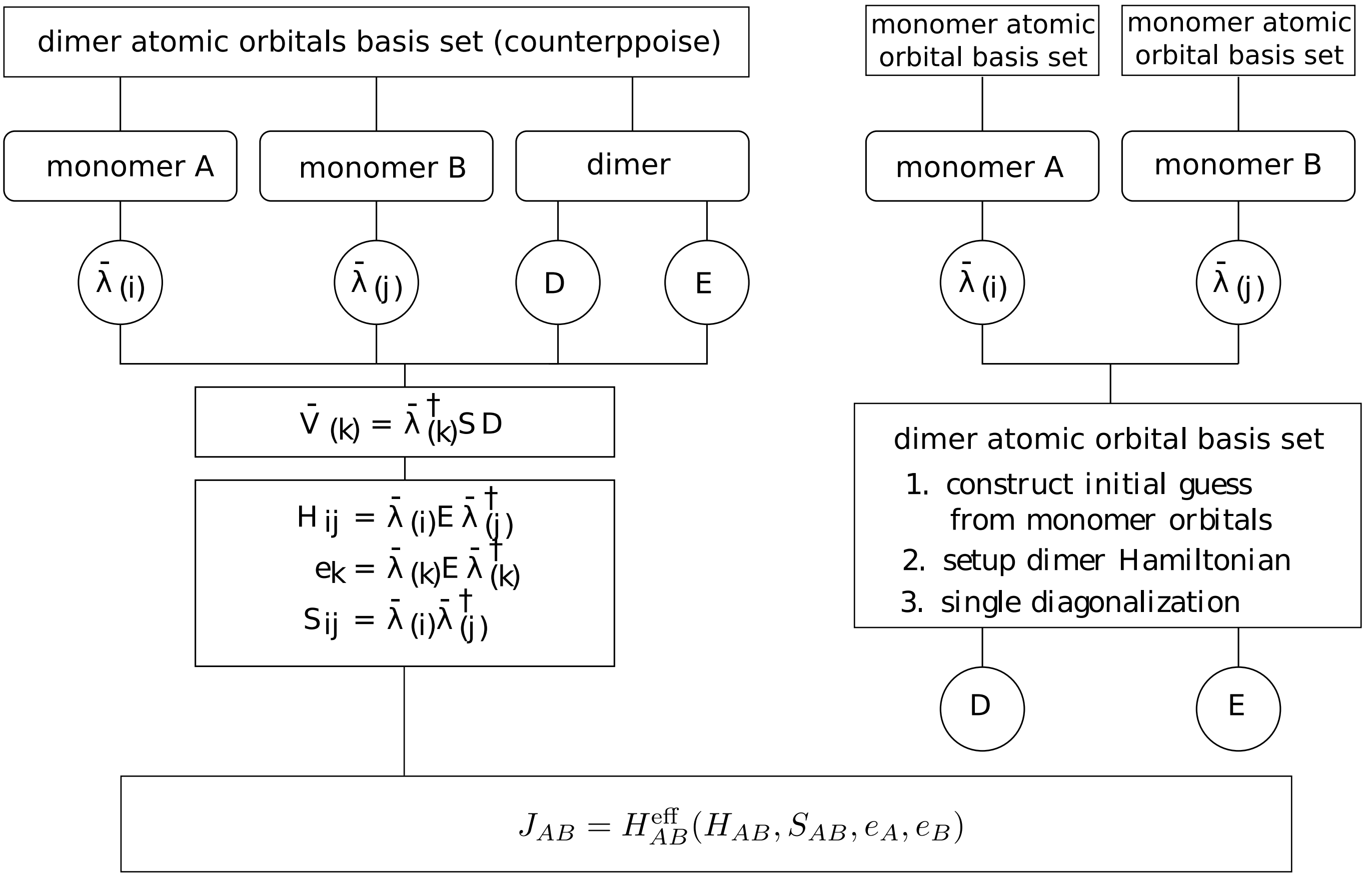

dimer atomic orbitals basis set (counterppoise)
monomer A
monomer B
dimer
$\bar{\lambda}$ (i)
$\bar{\lambda}$ (j)
D
E
$\bar{V}_{(k)} = \bar{\lambda}^{\dagger}_{(k)} S D$
$H_{ij} = \bar{\lambda}_{(i)} E \bar{\lambda}^{\dagger}_{(j)}$
$e_k = \bar{\lambda}_{(k)} E \bar{\lambda}^{\dagger}_{(k)}$
$S_{ij} = \bar{\lambda}_{(i)} \bar{\lambda}^{\dagger}_{(j)}$
monomer atomic orbital basis set
monomer atomic orbital basis set
monomer A
monomer B
$\bar{\lambda}$ (i)
$\bar{\lambda}$ (j)
dimer atomic orbital basis set
1. construct initial guess from monomer orbitals
2. setup dimer Hamiltonian
3. single diagonalization
D
E
$J_{AB} = H^{\text{eff}}_{AB}(H_{AB}, S_{AB}, e_A, e_B)$

Figure 5: Schematics of the DIPRO method. (a) General workflow of the projection technique. (b) Strategy of the efficient noCP+noSCF implementation, in which the monomer calculations are performed independently form the dimer configurations (noCP), using the `edft` calculator. The dimer Hamiltonian is subsequently constructed based on an initial guess formed from monomer orbitals and only diagonalized once (noSCF) before the transfer integral is calculated by projection. This second step is performed by the `idft` calculator.

quantum-chemical calculations for each individual monomer and one for the dimer to obtain this input data is extremely demanding. Several simplifications can be made to reduce the computational effort, such as using non-Counterpoise basis sets for the monomers (thereby decoupling the monomer calculations from the dimer run) and performing only a single SCF step in a dimer calculation starting from an initial guess formed from a superposition of monomer orbitals. This ”noCP+noSCF” variant of `DIPRO` is shown in figure 5(a) and recommended for production runs. A detailed comparative study of the different variants can be found in [5].

The code currently contains supports evaluation of transfer integrals from quantum-chemical calculations performed with the `Gaussian`, `Turbomole`, and `NWChem` packages. The interfacing procedure consists of three main steps: generation of input files for monomers and dimers, performing the actual quantum-chemical calculations, and calculating the transfer integrals.

Monomer calculations

First, hopping sites and a neighbor list need to be generated from the atomistic topology and trajectory and written to the `state.sql` file. Then the parallel `edft` calculator manages the calculation of the monomer properties required for the determination of electronic coupling elements. Specifically, the individual steps it performs are:

1. Creation of a job file containing the list of molecules to be calculated with DFT

Writing job file for `edft`

```
ctp_parallel -o options.xml -f state.sql -e edft -j write
```

2. Running of all jobs in job file

Running all `edft` jobs

```
ctp_parallel -o options.xml -f state.sql -e edft -j run
```

which includes

- creating the input files for the DFT calculation (using the package specified in `options.xml`) in the directory

  `OR_FILES/package/frame_F/mol_M`

  where `F` is the index of the frame in the trajectory, `M` is the index of a molecule in this frame,
- executing the DFT run, and
- after completion of this run, parsing the output (number of electrons, basis set, molecular orbital expansion coefficients), and saving it in compressed form to

  `OR_FILES/molecules/frame_F/molecule_M.orb`

Calculating the transfer integrals

After the momomer calculations have been completed successfully, the respective runs for dimers from the neighborlist can be performed using the parallel `idft` calculator, which manages the DFT runs for the hopping pairs and determines the coupling element using `DIPRO`. Again, several steps are required:

1. Creation of a job file containing the list of pairs to be calculated with DFT

Writing job file for `idft`

```
ctp_parallel -o options.xml -f state.sql -e idft -j write
```

2. Running of all jobs in job file

Running all `idft` jobs

```
ctp_parallel -o options.xml -f state.sql -e idft -j run
```

which includes

- creating the input files (including the merged guess for a noSCF calculation, if requested) for the DFT calculation (using the package specified in `options.xml`) in the directory

  `OR_FILES/package/frame_F/pair_M_N`

  where `M` and `N` are the indices of the molecules in this pair,
- executing the DFT run, and
- after completion of this run, parsing the output (number of electrons, basis set, molecular orbital expansion coefficients and energies, atomic orbital overlap matrix), and saving the pair information in compressed form to

  `OR_FILES/pairs/frame_F/pair_M_N.orb`

   - loading the monomer orbitals from the previously saved `*.orb files`.
   - calculating the coupling elements and write them to the job file

3. Reading the coupling elements from the job file and saving them to the `state.sql` file

Saving `idft` results from job file to `state.sql`

```
ctp_parallel -o options.xml -f state.sql -e idft -j read
```

Semiempirical

An approximate method based on Zerner's Intermediate Neglect of Differential Overlap (ZINDO) has been described in Ref. [7]. This semiempirical method is substantially faster than first-principles approaches, since it avoids the self-consistent calculations on each individual monomer and dimer. This allows to construct the matrix elements of the ZINDO Hamiltonian of the dimer from the weighted overlap of molecular orbitals of the two monomers. Together with the introduction of rigid segments, only a single self-consistent calculation on one isolated conjugated segment is required. All relevant molecular overlaps can then be constructed from the obtained molecular orbitals.

The main advantage of the molecular orbital overlap (MOO) library is *fast* evaluation of electronic coupling elements. Note that MOO is based on the ZINDO Hamiltonian which has limited applicability. The general advice is to first compare the accuracy of the MOO method to the DFT-based calculations.

MOO can be used both in a standalone mode and as an `izindo` calculator of VOTCA-CTP.

Since MOO constructs the Fock operator of a dimer from the molecular orbitals of monomers by translating and rotating the orbitals of rigid fragments, the optimized geometry of all conjugated segments and the coefficients of the molecular orbitals are required as its input in addition to the state file (`state.sql`) with the neighbor list. Coordinates are stored in `geometry.xyz` files with four columns, first being the atom type and the next three atom coordinates. This is a standard `xyz` format without a header. Note that the atom order in the `geometry.xyz` files can be different from that of the mapping files. The correspondence between the two is established in the `map.xml` file.

Attention

> Izindo requires the specification of orbitals for hole and electron transport in `map.xml`. They are the HOMO and LUMO respectively and can be retrieved from the `log` file from which the `zindo.orb` file is generated. The number of `alpha electrons` is the HOMO, the LUMO is HOMO+1

The calculated transfer integrals are immediately saved to the `state.sql` file.

Transfer integrals from `izindo`

```
ctp_run -o options.xml -f state.sql -e izindo
```

Rates

Charge transfer rates can be postulated based on intuitive physical considerations, as it is done in the Gaussian disorder models [24, 29–31]. Alternatively, charge transfer theories can be used to evaluate rates from quantum chemical calculations [3, 5, 12, 32–34]. In spite of being significantly more computationally demanding, the latter approach allows to link the chemical and electronic structure, as well as the morphology, to charge dynamics.

Classical charge transfer

The high temperature limit of classical charge transfer theory [35, 36] is often used as a trade-off between theoretical rigor and computational complexity. It captures key parameters which influence charge transport while at the same time providing an analytical expression for the rate. Within this limit, the transfer rate for a charge to hop from a site i to a site j reads

$$\omega_{ij} = \frac{2\pi}{\hbar} \frac{J_{ij}^2}{\sqrt{4\pi\lambda_{ij}k_{\mathrm{B}}T}} \exp\left[-\frac{(\Delta E_{ij}-\lambda_{ij})^2}{4\lambda_{ij}k_{\mathrm{B}}T}\right], \tag{31}$$

where T is the temperature, $\lambda_{ij} = \lambda_{ij}^{\mathrm{int}} + \lambda_{ij}^{\mathrm{out}}$ is the reorganization energy, which is a sum of intra- and inter-molecular (outersphere) contributions, ΔE_{ij} is the site-energy difference, or driving force, and J_{ij} is the electronic coupling element, or transfer integral.

Semi-classical charge transfer

The main assumptions in eq. (31) are non-adiabaticity (small electronic coupling and charge transfer between two diabatic, non-interacting states), and harmonic promoting modes, which are treated classically. At ambient conditions, however, the intramolecular promoting mode, which roughly corresponds to C-C bond stretching, has a vibrational energy of $\hbar\omega \approx 0.2\,\mathrm{eV} \gg k_{\mathrm{B}}T$ and should be treated quantum-mechanically. The outer-sphere (slow) mode has much lower vibrational energy than the intramolecular promoting mode, and therefore can be treated classically. The weak interaction between molecules also implies that each molecule has its own, practically independent, set of quantum mechanical degrees of freedom.

A more general, quantum-classical expression for a bimolecular multi-channel rate is derived in the Supporting Information of ref. [3] and has the following form

$$\omega_{ij} = \frac{2\pi}{\hbar} \frac{|J_{ij}|^2}{\sqrt{4\pi\lambda_{ij}^{\mathrm{out}}k_{\mathrm{B}}T}} \sum_{l',m'=0}^{\infty} |\langle\chi_{i0}^{c}|\chi_{il'}^{n}\rangle|^2 |\langle\chi_{j0}^{n}|\chi_{jm'}^{c}\rangle|^2 \exp\left\{-\frac{\left[\Delta E_{ij} - \hbar(l'\omega_i^n + m'\omega_j^c) - \lambda_{ij}^{\mathrm{out}}\right]^2}{4\lambda_{ij}^{\mathrm{out}}k_{\mathrm{B}}T}\right\}. \tag{32}$$

If the curvatures of intramolecular PES of charged and neutral states of a molecule are different, that is $\omega_i^c \neq \omega_i^n$, the corresponding reorganization energies, $\lambda_i^{cn} = \frac{1}{2}[\omega_i^n(q_i^n - q_i^c)]^2$ and $\lambda_i^{nc} = \frac{1}{2}[\omega_i^c(q_i^n - q_i^c)]^2$, will also differ. In this case the Franck-Condon (FC) factors for discharging of molecule i read [37]

$$|\langle\chi_{i0}^{c}|\chi_{il'}^{n}\rangle|^2 = \frac{2}{2^{l'}l'!}\frac{\sqrt{\omega_i^c\omega_i^n}}{(\omega_i^c+\omega_i^n)}\exp\left(-|s_i|\right)\left[\sum_{\substack{k=0\\k\text{ even}}}^{l'}\binom{l'}{k}\left(\frac{2\omega_i^c}{\omega_i^c+\omega_i^n}\right)^{k/2}\frac{k!}{(k/2)!}H_{l'-k}\left(\frac{s_i}{\sqrt{2S_i^{cn}}}\right)\right]^2, \tag{33}$$

where $H_n(x)$ is a Hermite polynomial, $s_i = 2\sqrt{\lambda_i^{nc}\lambda_i^{cn}}/\hbar(\omega_i^c+\omega_i^n)$, and $S_i^{cn} = \lambda_i^{cn}/\hbar\omega_i^c$. The FC factors for charging of molecule j can be obtained by substituting $(s_i, S_i^{cn}, \omega_i^c)$ with $(-s_j, S_j^{nc}, \omega_j^n)$. In order to evaluate the FC factors, the internal reorganization energy λ_i^{cn} can be computed from the intramolecular PES.

Numerical estimates show that if $\lambda_{ij}^{\mathrm{int}} \approx \lambda_{ij}^{\mathrm{out}}$ and $|\Delta E_{ij}| \ll \lambda_{ij}^{\mathrm{out}}$ the rates are similar to those of eq. (31). In general, there is no robust method to compute $\lambda_{ij}^{\mathrm{out}}$ [38] and both reorganization energies are often assumed to be of the same order of magnitude. In this case the second condition also holds, unless there are large differences in electron affinities or ionization potentials of neighboring molecules, e.g. in donor-acceptor blends.

To calculate rates of the type specified in `options.xml` for all pairs in the neighbor list and to save them into the `state.sql` file, run the `rates` calculator. Note that all required ingredients (reorganization energies, transfer integrals, and site energies have to be calculated before).

Calculation of transfer rates

```
ctp_run  -o options.xml  -f state.sql  -e rates
```

Master equation

Having determined the list of conjugated segments (hopping sites) and charge transfer rates between them, the next task is to solve the master equation which describes the time evolution of the system

$$\frac{\partial P_\alpha}{\partial t} = \sum_\beta P_\beta \Omega_{\beta\alpha} - \sum_\beta P_\alpha \Omega_{\alpha\beta}, \tag{34}$$

where P_α is the probability of the system to be in a state α at time t and $\Omega_{\alpha\beta}$ is the transition rate from state α to state β. A state α is specified by a set of site occupations, $\{\alpha_i\}$, where $\alpha_i = 1(0)$ for an occupied (unoccupied) site i, and the matrix $\hat{\Omega}$ can be constructed from rates ω_{ij}.

Kinetic Monte Carlo

The solution of eq. (34) is be obtained by using kinetic Monte Carlo (KMC) methods. KMC explicitly simulates the dynamics of charge carriers by constructing a Markov chain in state space and can find both stationary and transient solutions of the master equation. The main advantage of KMC is that only states with a direct link to the current state need to be considered at each step. Since these can be constructed solely from current site occupations, extensions to multiple charge carriers (without the mean-field approximation), site-occupation dependent rates (needed for the explicit treatment of Coulomb interactions), and different types of interacting particles and processes, are straightforward. To optimize memory usage and efficiency, a combination of the variable step size method [39] and the first reaction method is implemented.

To obtain the dynamics of charges using KMC, the program `kmc_run` executes a specific calculator after reading its options (charge carrier type, runtime, numer of carriers etc.) from `options.xml`.

KMC for a single carrier in periodic boundary conditions

```
kmc_run  -o options.xml  -f state.sql  -e kmcsingle
```

KMC for multiple carriers of the same type in periodic boundary conditions

```
kmc_run  -o options.xml  -f state.sql  -e kmcmultiple
```

Finite size effects

Predictions of charge-carrier mobilities in partially disordered semiconductors rely on charge transport simulations in systems which are only several nanometers thick. As a result, simulated charge transport might be dispersive for materials with large energetic disorder [40, 41] and simulated mobilities are system-size dependent. In time-of-flight (TOF) experiments, however, a typical sample thickness is in the micrometer range and transport is often nondispersive. In order to link simulation and experiment, one needs to extract the nondispersive mobility from simulations of small systems, where charge transport is dispersive at room temperature.

Such extrapolation is possible if the temperature dependence of the nondispersive mobility is known in a wide temperature range. For example, one can use analytical results derived for one-dimensional models [42–44]. The mobility-temperature dependence can then be parametrized by simulating charge transport at elevated temperatures, for which transport is nondispersive even for small system sizes. This dependence can then be used to extrapolate to the nondispersive mobility at room temperature [4].

For Alq_3, the charge carrier mobility of a periodic system of 512 molecules was shown to be more than three orders of magnitude higher than the nondispersive mobility of an infinitely large system [4]. Furthermore, it was shown that the transition between the dispersive and nondispersive transport has a logarithmic dependence on the number of hopping sites N. Hence, a brute-force increase of the system size cannot resolve the problem for compounds with large energetic disorder σ, since N increases exponentially with σ^2.

Macroscopic observables

Spatial distributions of charge and current densities can provide a better insight in the microscopic mechanisms of charge transport. If O is an observable which has a value O_α in a state α, its ensemble average at time t is a sum over all states weighted by the probability P_α to be in a state α at time t

$$\langle O \rangle = \sum_\alpha O_\alpha P_\alpha . \tag{35}$$

If O does not explicitly depend on time, the time evolution of $\langle O \rangle$ can be calculated as

$$\frac{d\langle O \rangle}{dt} = \sum_{\alpha,\beta} \left[P_\beta \Omega_{\beta\alpha} - P_\alpha \Omega_{\alpha\beta} \right] O_\alpha = \sum_{\alpha,\beta} P_\beta \Omega_{\beta\alpha} \left[O_\alpha - O_\beta \right] . \tag{36}$$

If averages are obtained from KMC trajectories, $P_\alpha = s_\alpha / s$, where s_α is the number of Markov chains ending in the state α after time t, and s is the total number of chains.

Alternatively, one can calculate time averages by analyzing a single Markov chain. If the total occupation time of the state α is τ_α then

$$\overline{O} = \frac{1}{\tau} \sum_\alpha O_\alpha \tau_\alpha \, , \tag{37}$$

where $\tau = \sum_\alpha \tau_\alpha$ is the total time used for time averaging.

For ergodic systems and sufficient sampling times, ensemble and time averages should give identical results. In many cases, the averaging procedure reflects a specific experimental technique. For example, an ensemble average over several KMC trajectories with different starting conditions corresponds to averaging over injected charge carriers in a time-of-flight experiment. In what follows, we focus on the single charge carrier (low concentration of charges) case.

Charge density

For a specific type of particles, the microscopic charge density of a site i is proportional to the occupation probability of the site, p_i

$$\rho_i = e p_i / V_i \, , \tag{38}$$

where, for an irregular lattice, the effective volume V_i can be obtained from a Voronoi tessellation of space. For reasonably uniform lattices (uniform site densities) this volume is almost independent of the site and a constant volume per cite, $V_i = V/N$, can be assumed. In the macroscopic limit, the charge density can be calculated using a smoothing kernel function, i.e. a distance-weighted average over multiple sites. Site occupations p_i can be obtained from eq. (35) or eq. (37) by using the occupation of site i in state α as an observable.

If the system is in thermodynamic equilibrium, that is without sources or sinks and without circular currents (and therefore no net flux) a condition, known as detailed balance, holds

$$p_j \omega_{ji} = p_i \omega_{ij}, \tag{39}$$

It can be used to test whether the system is ergodic or not by correlating $\log p_i$ and the site energy E_i. Indeed, if $\lambda_{ij} = \lambda_{ji}$ the ratios of forward and backward rates are determined solely by the energetic disorder, $\omega_{ji}/\omega_{ij} = \exp(-\Delta E_{ij}/k_\mathrm{B}T)$ (see eq. (31)).

Current

If the position of the charge, $\vec{r}$, is an observable, the time evolution of its average $\langle \vec{r} \rangle$ is the total current in the system

$$\vec{J} = e \langle \vec{v} \rangle = e \frac{d \langle \vec{r} \rangle}{dt} = e \sum_{i,j} p_j \omega_{ji} (\vec{r}_i - \vec{r}_j). \tag{40}$$

Symmetrizing this expression we obtain

$$\vec{J} = \frac{1}{2} e \sum_{i,j} \left(p_j \omega_{ji} - p_i \omega_{ij} \right) \vec{r}_{ij}, \tag{41}$$

where $\vec{r}_{ij} = \vec{r}_i - \vec{r}_j$. Symmetrization ensures equal flux splitting between neighboring sites and absence of local average fluxes in equilibrium. It allows to define a local current through site i as

$$\vec{J}_i = \frac{1}{2} e \sum_{j} \left(p_j \omega_{ji} - p_i \omega_{ij} \right) \vec{r}_{ij}. \tag{42}$$

A large value of the local current indicates that the site contributes considerably to the total current. A collection of such sites thus represents most favorable charge pathways [45].

Mobility and diffusion constant

For a single particle, e.g. a charge or an exciton, a zero-field mobility can be determined by studying particle diffusion in the absence of external fields. Using the particle displacement squared, $\Delta \boldsymbol{r}_i^2$, as an observable we obtain

$$2dD_{\gamma\delta} = \frac{d \langle \Delta r_{i,\gamma} \Delta r_{i,\delta} \rangle}{dt} = \sum_{\substack{i,j \\ i \neq j}} p_j \omega_{ji} \left(\Delta r_{i,\gamma} \Delta r_{i,\delta} - \Delta r_{j,\gamma} \Delta r_{j,\delta} \right) = \sum_{\substack{i,j \\ i \neq j}} p_j \omega_{ji} \left(r_{i,\gamma} r_{i,\delta} - r_{j,\gamma} r_{j,\delta} \right) . \tag{43}$$

Here $\vec{r}_i$ is the coordinate of the site i, $D_{\gamma\delta}$ is the diffusion tensor, $\gamma, \delta = x, y, z$, and $d = 3$ is the system dimension. Using the Einstein relation,

$$D_{\gamma\delta} = k_\mathrm{B} T \mu_{\gamma\delta} \, , \tag{44}$$

one can, in principle, obtain the zero-field mobility tensor $\mu_{\gamma\delta}$. Eq. (43), however, does not take into account the use of periodic boundary conditions when simulating charge dynamics. In this case, the simulated occupation probabilities can be compared to the solution of the Smoluchowski equation with periodic boundary conditions (see the supporting information for details).

Alternatively, one can directly analyze time-evolution of the KMC trajectory and obtain the diffusion tensor from a linear fit to the mean square displacement, $\overline{\Delta r_{i,\gamma} \Delta r_{i,\delta}} = 2dD_{\gamma\delta} t$.

The charge carrier mobility tensor, $\hat{\mu}$, for any value of the external field can be determined either from the average charge velocity defined in eq. (40)

$$\langle \vec{v} \rangle = \sum_{i,j} p_j \omega_{ji} (\vec{r}_i - \vec{r}_j) = \hat{\mu} \vec{F} \, , \tag{45}$$

or directly from the KMC trajectory. In the latter case the velocity is calculated from the unwrapped (if periodic boundary conditions are used) charge displacement vector divided by the total simulation time. Projecting this velocity on the direction of the field $\vec{F}$ yields the charge carrier mobility in this particular direction. In order to improve statistics, mobilities can be averaged over several KMC trajectories and MD snapshots.

Spatial correlations of energetic disorder

Long-range, e.g. electrostatic and polarization, interactions often result in spatially correlated disorder [46], which affects the onset of the mobility-field (Poole-Frenkel) dependence [42, 47, 48].

To quantify the degree of correlation, one can calculate the spatial correlation function of E_i and E_j at a distance r_{ij}

$$C(r_{ij}) = \frac{\langle (E_i - \langle E \rangle)(E_j - \langle E \rangle) \rangle}{\langle (E_i - \langle E \rangle)^2 \rangle}, \tag{46}$$

where $\langle E \rangle$ is the average site energy. $C(r_{ij})$ is zero if E_i and E_j are uncorrelated and 1 if they are fully correlated. For a system of randomly oriented point dipoles, the correlation function decays as $1/r$ at large distances [49].

For systems with spatial correlations, variations in site energy differences, ΔE_{ij}, of pairs of molecules from the neighbor list are smaller than variations in site energies, E_i, of all individual molecules. Since only neighbor list pairs affect transport, the distribution of ΔE_{ij} rather than that of individual site energies, E_i, should be used to characterize energetic disorder.

Note that the `eanalyze` calculator takes into account *all* contributions to the site energies

Analyze distribution and correlations of site energeies

```
ctp_run -o options.xml -f state.sql -e eanalyze
```

Input and output files

Molecular orbitals

If the semi-empirical method is used to calculate electronic coupling elements, molecular orbitals of all molecules must be supplied. They can be generated using `Gaussian` program. The `Gaussian` input file for DCV2T is shown in listing 3. Provided with this input, `Gaussian` will generate `fort.7` file which contains the molecular orbitals of a DCV2T. This file can be renamed to `DCV2T.orb`. Note that the order of the atoms in the input file and the order of coefficients should always match. Therefore, the coordinate part of the input file must be supplied together with the orbitals. We will assume the coordinates, in the format `atom_type:  x y z`, is saved to the `DCV2T.xyz` file.

Attention

Izindo requires the specification of orbitals for hole and electron transport in `map.xml`. They are the HOMO and LUMO respectively and can be retrieved from the `log` file from which the `DCV2T.orb` file is generated. The number of `alpha electrons` is the HOMO, the LUMO is HOMO+1

Listing 3: `Gaussian` input file `get_orbitals.com` used for generating molecular orbitals. The first line contains the name of the check file, the second the requested RAM. `int=zindos` requests the method ZINDO, `punch=mo` states that the molecular orbitals ought to be written to the `fort.7` file, `nosymm` forbids use of symmetry and is necessary to ensure correct position of orbitals with respect to the provided coordinates. The two integer numbers correspond to the charge and multiplicity of the system: `0 1` corresponds to a neutral system with a multiplicity of one. They are followed by the types and coordinates of all atoms in the molecule.

```
%chk=DCV2T.chk
%mem=100Mb
#p int=zindos punch=mo nosymm

DCV2T molecular orbitals

0 1
S         -1.44650        2.12185        0.00135
C         -2.43098        0.58936       -0.00048
C         -1.59065       -0.51859       -0.00146
C         -0.21222       -0.22233       -0.00095
C          0.07761        1.13376        0.00040
S          2.87651        0.79316        0.00148
C          3.86099        2.32565        0.00235
C          3.02066        3.43359        0.00231
C          1.64223        3.13733        0.00162
C          1.35240        1.78125        0.00114
C         -3.85350        0.52245       -0.00081
C         -4.79569        1.52479       -0.00008
C         -6.18500        1.18622       -0.00117
C         -4.47544        2.91565        0.00081
C          5.28350        2.39256        0.00296
C          6.22569        1.39020        0.00327
C          7.61500        1.72876        0.00432
C          5.90542       -0.00064        0.00333
N         -7.32389        0.89743       -0.00195
N         -4.21872        4.06274        0.00142
N          8.75389        2.01754        0.00510
N          5.64864       -1.14772        0.00361
H         -1.98064       -1.52966       -0.00256
H          0.55785       -0.98374       -0.00169
H          3.41065        4.44466        0.00272
H          0.87216        3.89874        0.00147
H         -4.24640       -0.49192       -0.00188
H          5.67641        3.40692        0.00337
```

Monomer calculations for DFT transfer integrals

Listing 4: Example `package.xml` file for the `Gaussian` package required in the options of the `edft` calculator for the monomer calculations as preparation for the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>gaussian</name>
  <executable>g09</executable>
  <checkpoint></checkpoint>
  <scratch></scratch>

  <charge>0</charge>
  <spin>1</spin>
  <options># pop=minimal pbepbe/6-311g** scf=tight punch=mo nosymm test</options>
  <memory>1Gb</memory>
  <threads>2</threads>

  <cleanup></cleanup>
```

```
</package>
```

Listing 5: Example `package.xml` file for the `Turbomole` package required in the options of the `edft` calculator for the monomer calculations as preparation for the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>turbomole</name>
  <executable>ridft</executable>
  <scratch>/tmp</scratch>

  <options>
TITLE
a coord
*
no
b all def-TZVP
*
eht
y
0
y
dft
on
func
pbe
grid
m3
*
ri
on
m 300
*
scf
conv
7
iter
200

marij

q
  </options>

  <cleanup></cleanup>
</package>
```

Listing 6: Example `package.xml` file for the `NWChem` package required in the options of the `edft` calculator for the monomer calculations as preparation for the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>nwchem</name>
  <executable>nwchem</executable>
  <checkpoint></checkpoint>
  <scratch>/tmp/nwchem</scratch>
  <charge>0</charge>
  <spin>1</spin>
  <threads>1</threads>
  <memory></memory>
  <options>
start
basis
 * library 6-311gss
end
```

```
memory 1500 mb

dft
 xc xpbe96 cpbe96
 direct
 iterations 100
 noprint "final vectors analysis"
end
task dft
</options>
  <cleanup></cleanup>
</package>
```

Pair calculations for DFT transfer integrals

Listing 7: Example `package.xml` file for the `Gaussian` package required in the options of the `idft` calculator for the pair calculations and the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>gaussian</name>
  <executable>g09</executable>
  <checkpoint></checkpoint>
  <scratch></scratch>

  <charge>0</charge>
  <spin>1</spin>
  <options># pop=minimal pbepbe/6-311g** nosymm  IOp(3/33=1,3/36=-1) punch=mo guess=cards scf=(maxcycle
  <memory>1Gb</memory>
  <threads>1</threads>

  <cleanup></cleanup>
</package>
```

Listing 8: Example `package.xml` file for the `Turbomole` package required in the options of the `idft` calculator for the pair calculations and the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>turbomole</name>
  <executable>ridft</executable>
  <scratch>/tmp</scratch>

  <options>
$intsdebug cao
a coord
*
no
b all def-TZVP
*
eht
y
0
y
dft
on
func
pbe
grid
m3
*
ri
on
```

```
m 300
*
scf
conv
7
iter
1
diis
3
damp
0.00



marij

q
  </options>

  <cleanup></cleanup>
</package>
```

Listing 9: Example `package.xml` file for the `NWChem` package required in the options of the `idft` calculator for the pair calculations and the determination of transfer integrals using `DIPRO`.

```
<package>
  <name>nwchem</name>
  <executable>nwchem</executable>
  <checkpoint></checkpoint>
  <scratch>/tmp/nwchem</scratch>
  <charge>0</charge>
  <spin>1</spin>
  <memory></memory>
  <threads>1</threads>
  <options>
start
basis
 * library 6-311gss
end
memory 1500 mb

dft
 print "ao overlap"
 xc xpbe96 cpbe96
 direct
 iterations 1
 convergence nodamping nodiis
 noprint "final vectors analysis"
 vectors input system.movecs
end
task dft
</options>
  <cleanup></cleanup>
</package>
```

DFT transfer integrals

Listing 10: Example `TI.xml` file created as the output of a `DIPRO` calculation. Due to slightly different implementations, the orbitals indices refer to monomer indices in a `Gaussian` run but to indices in the merged dimer guess in a `Turbomole` run.

```
<pair name="pair_100_155">
   <parameters>
      <HOMO_A>162</HOMO_A>
      <NoccA>1</NoccA>
      <LUMO_A>164</LUMO_A>
      <NvirtA>1</NvirtA>
      <HOMO_B>161</HOMO_B>
      <NoccB>1</NoccB>
      <LUMO_B>163</LUMO_B>
      <NvirtB>1</NvirtB>
   </parameters>
    <transport name="hole">
        <channel name="single">
            <J>1.546400416750696E-003</J>
            <e_A>-6.30726450715697</e_A>
            <e_B>-6.36775613794166</e_B>
        </channel>
        <channel name="multi">
           <molecule name="A">
               <e_HOMOm0>-6.30726450715697</e_HOMOm0>
           </molecule>
           <molecule name="B">
               <e_HOMOm0>-6.36775613794166</e_HOMOm0>
           </molecule>
               <dimer name="integrals">
                    <T_00>1.546400416750696E-003</T_00>
                    <J_sq_degen>2.391354248926727E-006</J_sq_degen>
                    <J_sq_boltz>2.391354248926727E-006</J_sq_boltz>
               </dimer>
        </channel>
    </transport>
    <transport name="electron">
        <channel name="single">
            <J>-2.797473760331286E-003</J>
            <e_A>-4.50318366770689</e_A>
            <e_B>-4.53143397059021</e_B>
        </channel>
        <channel name="multi">
               <molecule name="A">
                    <e_LUMOp0>-4.50318366770689</e_LUMOp0>
               </molecule>
               <molecule name="B">
                    <e_LUMOp0>-4.53143397059021</e_LUMOp0>
               </molecule>
               <dimer name="integrals">
                    <T_00>-2.797473760331286E-003</T_00>
                    <J_sq_degen>7.825859439742066E-006</J_sq_degen>
                    <J_sq_boltz>7.825859439742066E-006</J_sq_boltz>
               </dimer>
        </channel>
    </transport>
</pair>
```

State file

All data structures are saved to the `state.sql` file in sqlite3 format, see http://www.sqlite.org/. They are available in form of tables in the `state.sql` file as can be seen by the command
`sqlite3 state.sql " .tables "`

An example of such a table are `molecules`. The full table can be displayed using the command (similar for the other tables)

```
sqlite3 state.sql " SELECT * FROM molecules "
```

The meaning of all the entries in the table can be displayed by a command like

```
sqlite3 state.sql " .SCHEMA molecules "
```

The first and second entry are integers for internal and regular id of the molecule and the third entry is the name. A single field from the table like the name of the molecule can be displayed by a command like

```
sqlite3 state.sql " SELECT name FROM molecules "
```

Besides `molecules`, the following tables are stored in the `state.sql`:

`conjseg_properties`:

Conjugated segments are stored with id, name and x,y,z coordinates of the center of mass in nm.

`conjsegs`:

Reorganization energies for charging or discharging a conjugated segment are stored together with the coulomb energy and any other user defined energy contribution (in eV) and occupation probabilities.

`pairs`:

The pairs from the neighborlist are stored with the pair id, the id of the first and second segment, the rate from the first to the second , the rate from the second to the first (both in s^{-}1) and the x,y,z coordinates in nm of the distance between the first and the second segment.

`pairintegrals`:

Transfer integrals for all pairs are stored in the following way: The pair id , the number for counting possible different electronic overlaps (e.g if only the frontier orbitals are taken into account this is always zero, while an effective value is stored in addition to the different overlaps of e.g. HOMO-1 and HOMO-1 if more frontier orbitals are taken into account) and the integral in eV.

`pairproperties`:

The outer sphere reorganization energy of all pairs is stored by an id, the pair id, a string `lambda_outer` and the energy in eV.

`conjsegs`:

Conjugated segments are saved in the following way: The id, the name, the type, the molecule id, the time frame, the x,y,z coordinates in nm and the occupation probability.

`conjseg_properties`:

Properties of the conjugated segments like reorganization energies for charging or discharging a charge unit or the coulomb contribution to the site energy are stored by: id, conjugated segment id, a string like `lambda_intra_charging`, `lambda_intra_discharging` or `energy_coulomb` and a corresponding value in eV.

The tables `rigidfrag_properties`, `rigidfrags` and `frames` offer information about rigid fragments and time frames including periodic boundary conditions.

The data in the `state.sql` file can also be modified by the user. Here is an example how to modify the transfer integral between the conjugated segments number one and two assuming that they are in the neighborlist. Their pair id can be found by the command

```
pair_ID=`sqlite3 state.sql "SELECT _id FROM pairs WHERE conjseg1=1 AND conjseg2=2"`
```

The old value of the transfer integral can be deleted using

```
sqlite3 state.sql "DELETE FROM pair_integrals WHERE pair=$pair_ID"
```

Finally the new transfer integral J can be written to the `state.sql` file by the command

```
sqlite3 state.sql "INSERT INTO pair_integrals (pair,num,J) VALUES ($pair_ID,0,$J)"
```

Here the `num=0` indicates that only the effective transfer integrals is written to the file, while other values of `num` would correspond to overlap between other orbitals than the frontier orbitals.

In a similar way the coulomb contribution to the site energy of the first conjugated segment can be overwritten by first getting its id

```
c_ID=`sqlite3 state.sql "SELECT _id from conjseg_properties where conjseg=1 AND key =\"energy_
```

Then deleting the old value

```
sqlite3 state.sql "DELETE FROM from conjseg_properties WHERE _id=$c_ID"
```

Then the new coulomb energy E can be written to this id

```
sqlite3 state.sql "INSERT INTO conjseg_properties (_id,conjseg,key,value)
```

```
VALUES ($c_ID,1,\"energy_coulomb\",$E)"
```

Finally the resulting coulomb contribution to all conjugated segments can be displayed by

```
sqlite3 state.sql "SELECT * from conjseg_properties WHERE key=\"energy_coulomb\""
```

Programs

Programs execute specific tasks (calculators).

ctp_map

Generates QM|MD topology

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `-t [ --topology ] arg` topology
- `-c [ --coordinates ] arg` coordinates or trajectory
- `-s [ --segments ] arg` definition of segments and fragments
- `-f [ --file ] arg` state file
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages

ctp_dump

Extracts information from the state file

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `-o [ --options ] arg` calculator options
- `-f [ --file ] arg` sqlight state file, *.sql
- `-i [ --first-frame ] arg (=1)` start from this frame
- `-n [ --nframes ] arg (=1)` number of frames to process
- `-t [ --nthreads ] arg (=1)` number of threads to create
- `-s [ --save ] arg (=1)` whether or not to save changes to state file
- `-e [ --extract ] arg` List of extractors separated by ',' or ' '
- `-l [ --list ]` Lists all available extractors
- `-d [ --description ] arg` Short description of an extractor
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages

ctp_tools

Runs charge transport tools

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `-t [ --nthreads ] arg (=1)` number of threads to create
- `-o [ --options ] arg` calculator options
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages
- `-e [ --execute ] arg` List of tools separated by ',' or ' '
- `-l [ --list ]` Lists all available tools
- `-d [ --description ] arg` Short description of a tool

ctp_run

Runs charge transport calculators

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `-o [ --options ] arg` calculator options
- `-f [ --file ] arg` sqlight state file, *.sql
- `-i [ --first-frame ] arg (=1)` start from this frame
- `-n [ --nframes ] arg (=1)` number of frames to process
- `-t [ --nthreads ] arg (=1)` number of threads to create
- `-s [ --save ] arg (=1)` whether or not to save changes to state file
- `-e [ --execute ] arg` List of calculators separated by ’,’ or ’ ’
- `-l [ --list ]` Lists all available calculators
- `-d [ --description ] arg` Short description of a calculator
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages

ctp_parallel

Runs job-based heavy-duty calculators

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `-o [ --options ] arg` calculator options
- `-f [ --file ] arg` sqlite state file, *.sql
- `-i [ --first-frame ] arg (=1)` start from this frame
- `-n [ --nframes ] arg (=1)` number of frames to process
- `-t [ --nthreads ] arg (=1)` number of threads to create
- `-s [ --save ] arg (=1)` whether or not to save changes to state file
- `-r [ --restart ] arg` restart pattern: ’host(pc1:234) stat(FAILED)’
- `-c [ --cache ] arg (=8)` assigns jobs in blocks of this size
- `-j [ --jobs ] arg (=run)` task(s) to perform: input, run, import
- `-m [ --maxjobs ] arg (=-1)` maximum number of jobs to process (-1 = inf)
- `-e [ --execute ] arg` List of calculators separated by ’,’ or ’ ’
- `-l [ --list ]` Lists all available calculators
- `-d [ --description ] arg` Short description of a calculator
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages

moo_overlap

- `-h [ --help ]` display this help and exit
- `-v [ --verbose ]` be loud and noisy
- `--man` output man-formatted manual pages
- `--tex` output tex-formatted manual pages
- `--conjseg arg` xml file describing two conjugated segments
- `--pos1 arg` position and orientation of molecule 1
- `--pos2 arg` position and orientation of molecule 2
- `--pdb arg (=geometry.pdb)` pdb file of two molecules

kmc_run

Runs specified calculators

- `-h [ --help ]` display this help and exit

`-v [ --verbose ]` be loud and noisy
`-o [ --options ] arg` program and calculator options
`-f [ --file ] arg` sqlite state file
`-t [ --textfile ] arg` output text file (otherwise: screen output)
`-e [ --execute ] arg` list of calculators separated by commas or spaces
`-l [ --list ]` lists all available calculators
`-d [ --description ] arg` detailed description of a calculator
`--man` output man-formatted manual pages
`--tex` output tex-formatted manual pages

Calculators

Calculator is a piece of code which computes specific system properties, such as site energies, transfer integrals, etc. `ctp_run`, `kmc_run` are wrapper programs which executes such calculators. The generic syntax is

```
ctp_run -e "calc1, calc2, ..." -o options.xml
```

File `options.xml` lists all options needed to run a specific calculator. The format of this file is explained in listing 11. A complete list of calculators is given in the calculators reference section.

Listing 11: A part of the `options.xml` file with options for the `calculator_name{1,2}` calculators.

```
<calculator_name1>
          <option1>value1</option1>
          <option2>value2</option2>
          ...
</calculator_name1>

<calculator_name2>
          <option1>value1</option1>
          <option2>value2</option2>
          ...
</calculator_name2>
...
```

A list of all calculators and their short descriptions can be obtain using

```
ctp_run --list
```

A detailed description of all options of a specific calculator(s) is available via

```
ctp_run --desc calc1,calc2,...
```

coupling

Electronic couplings from log and orbital files (GAUSSAIN, TURBOMOLE, NWChem)

option	default	unit	description
package			First-principles package
output	coupling.out.x		Output file
degeneracy	0	eV	Criterium for the degeneracy of two levels
moleculeA			
log	A.log		Log file of molecule A
orbitals	A.orb		Orbitals file
levels	3		Output HOMO, ..., HOMO-levels; LUMO, ..., LUMO+levels
trim	2		

moleculeB			
log	B.log		Log file of molecule B
orbitals	B.orb		Orbitals file
levels	3		Output HOMO, ..., HOMO-levels; LUMO, ..., LUMO+levels
trim	2		
dimerAB			
log	AB.log		Log file of dimer AB
orbitals	A.orb		Orbitals file

Return to the description of `coupling`.

log2mps

Generates an mps-file (with polar-site definitions) from a QM log-file

option	default	unit	description
package			QM package
logfile			Log-file generated by QM package, with population/esp-fit data

Return to the description of `log2mps`.

molpol

Molecular polarizability calculator (and optimizer)

option	default	unit	description
mpsfiles			
input			mps input file
output			mps output file
polar			xml file with infos on polarizability tensor
induction			
expdamp			Thole sharpness parameter
wSOR			mixing factor for convergence
maxiter			maximum number of iterations
tolerance			rel. tolerance for induced moments
target			
optimize			if 'true', refine atomic polarizabilities to match molecular polarizable volume specified in target.molpol
molpol			target polarizability tensor in format xx xy xz yy yz zz (this should be in the eigen-frame, hence xy = xz = yz = 0), if optimize=true the associated polarizable volume will be matched iteratively and the resulting set of polar sites written to mpsfiles.output
tolerance			relative tolerance when optimizing the polarizable volume

Return to the description of `molpol`.

pdb2map

Converts MD + QM files to VOTCA mapping. Combinations: pdb+xyz,gro+xyz,pdb

option	default	unit	description
pdb	conf.pdb		Input pdb file
gro	conf.gro		Input gro file

xyz	conf.xyz		Input xyz file
xml	conf.xml		Resulting xml file

Return to the description of `pdb2map`.

pdb2top

Generates fake Gromacs topology file .top

option	default	unit	description
num	1		Num of mols in the box
pdb	conf.pdb		Input pdb file
gro	conf.gro		Input gro file

Return to the description of `pdb2top`.

ptopreader

Reads binary .ptop-files (serialized from ewdbgpol) and processes them into something readable

option	default	unit	description
ptop_file			Binary archive .ptop-file

Return to the description of `ptopreader`.

eanalyze

Histogram and correlation function of site energies and pair energy differences

option	default	unit	description
resolution_sites		eV	Bin size for site energy histogram
resolution_pairs		eV	Bin size for pair energy histogram
resolution_space		eV	Bin size for site energy correlation
states			?

Return to the description of `eanalyze`.

eimport

Imports site energies from the output file of emultipole and writes them to the state file

option	default	unit	description

Return to the description of `eimport`.

einternal

Reads in site and reorganosation energies and writes them to the state file

option	default	unit	description
energiesXML			XML input file with vacuum site, reorganization (charging, discharging) energies

Return to the description of `einternal`.

emultipole

Evaluates polarization contribution based on the Thole model

option	default	unit	description
multipoles			Polar Site Definitions in GDMA punch-file format
control			Control options for induction computation
induce	1		Enter '1' / '0' to toggle induction on / off
first			First segment for which to compute site energies
last			Last segment for which to compute site energies
output			File to write site energies to. Site energies are also stored in the state file
check			Check mapping of polar sites to fragment
tholeparam			Thole parameters required for charge-smearing
cutoff		nm	Cut-off beyond which all interactions are neglected
cutoff2		nm	Cut-off beyond which polarization is neglected
expdamp			Damping exponent used in exponential damping function
scaling			1-n interaction scaling, currently not in use
esp			Control options for potential calculation
calcESP			Enter '1' / '0' to toggle on / off. If '1', site energies will not be evaluated
cube			
grid			XYZ file specifying grid points for potential evaluation
output			File to write grid-point potential to
esf			Control options for field calculation
calcESF			Enter '1' / '0' to toggle on / off. If '1', site energies will not be evaluated
grid			XYZ file specifying grid points for field evaluation
output			File to write grid-point field to
alphamol			Control options for molecular-polarizability calculation
calcAlpha			Enter '1' / '0' to toggle on / off. If '1', site energies will not be evaluated
output			File to write polarizability tensor in global frame and in diagonal form to
convparam			Convergence parameters for self-consistent field calculation
wSOR_N			Mixing factor for successive overrelaxation of neutral system, usually between 0.3 and 0.5
wSOR_C			Mixing factor for successive overrelaxation of charged system, usually between 0.3 and 0.5
tolerance			Convergence criterion, fulfilled if relative change smaller than tolerance
maxiter			Maximum number of iterations in the convergence loop

Return to the description of `emultipole`.

eoutersphere

Evaluates outersphere reorganization energy

option	default	unit	description
multipoles			XML allocation polar sites

method			Type of the method: **constant** - all pairs have value **lambda**. **spheres** - molecules are treated as spheres with radii **radius** and Pekar factor **pekar**. **dielectric** - with Pekar factor **pekar** and partial charges from resulting dielectric fields
lambdaconst		eV	The value for all pairs in the **constant** method
pekar			Pekar factor used for methods **spheres** and **dielectric**
segment			
type			
radius			
segment			
type			
radius			
cutoff		nm	Cutoff radius in between pair and the exterior molecule. Can be used in **spheres** and **dielectric**

Return to the description of `eoutersphere`.

ianalyze

Evaluates a histogram of a logarithm of squared couplings

option	default	unit	description
resolution_logJ2			Bin size of histogram log(J2)
states			States for which to calculate the histogram. Example: 1 -1

Return to the description of `ianalyze`.

iimport

Imports electronic couplings from xml of ctp-dipro using folders of pairdump

option	default	unit	description
idft_jobs_file			idft jobs file

Return to the description of `iimport`.

izindo

Semiempirical electronic coupling elements for all neighbor list pairs

option	default	unit	description
orbitalsXML			File with paths to .orb files

Return to the description of `izindo`.

jobwriter

Writes list of jobs for a parallel execusion

option	default	unit	description
keys			job type
states	n e h		hole, electron, nuetral: mps file is required
single_id			Segment ID as argument for mps.single

kmc_cutoff		nm	Pair-interaction cut-off as argument for mps.kmc

Return to the description of `jobwriter`.

neighborlist

Constructs a list of neighboring conjugated segments

option	default	unit	description
constant	0.5	nm	If provided, this value is used for all segment types
segments			A pair of segment types
type			Types of two segments. For types A and B this can be A A, A B or B B
cutoff		nm	Cutoff radius for centers of mass of rigid fragments
fragments	*		list of active fragments
file			File with the pair list: pairID seg1ID seg2ID seg1Type seg2Type

Return to the description of `neighborlist`.

pairdump

Coordinates of molecules and pairs from the neighbor list

option	default	unit	description
molecules			If **true** outputs single molecules, otherwise only pairs

Return to the description of `pairdump`.

profile

Density and site energy profiles

option	default	unit	description
axis			Axis along which to calculate density and energy profiles
direction	0 0 1		Axis direction
min		nm	Minimal projected position for manual binning
max		nm	Maximal projected position for manual binning
bin	0.1	nm	Spatial resolution of the profile
auto	1		’0’ for manual binning using min and max, ’1’ for automated
particles			
type	segments		What centers of mass to use: ’segments’ or ’atoms’
first	1		ID of the first segment
last	-1		ID of the last segment, -1 is the list end
output			
density	density.dat		Density profile file
energy	energy.dat		Energy profile file

Return to the description of `profile`.

rates

Hopping rates using classical or semi-classical expression

option	default	unit	description
field			Field in x y z direction
temperature		K	Temperature for rates
method			Method chosen to compute rates. Can either be **marcus** or **jortner**. The first is the high temperature limit of Marcus theory, the second is the rate proposed by Jortner and Bixon
nmaxvib	20		If the method of choice is **jortner**, the maximal number of excited vibrations on the molecules has to be specified as an integer for the summation
omegavib	0.2	eV	If the method of choice is **jortner**, the vibration frequency of the quantum mode has to be given in units of eV. The default value is close to the CC bond-stretch at 0.2eV

Return to the description of `rates`.

sandbox

Sandbox to test ctp classes

option	default	unit	description
ID			Not in use

Return to the description of `sandbox`.

stateserver

Export SQLite file to human readable format

option	default	unit	description
out			Output file name
pdb			PDB coordinate file name
keys			Sections to write to readable format (topology, segments, pairs, coordinates)

Return to the description of `stateserver`.

tdump

Coarse-grained and back-mapped (using rigid fragments) trajectories

option	default	unit	description
md	MD.pdb		Name of the coarse-grained trajectory
qm	QM.pdb		Name of the trajectory with back-substituted rigid fragments
frames	1		Number of frames to output

Return to the description of `tdump`.

vaverage

Computes site-centered velocity averages from site occupancies

option	default	unit	description
carriers			Carrier types for which to compute velocity averages
tabulate			Tabulate ’atoms’ or ’segments’

Return to the description of `vaverage`.

zmultipole

Evaluates polarization contribution based on the Thole model

option	default	unit	description
multipoles			Polar Site Definitions in GDMA punch-file format
control			Control options for induction computation
induce	1		Enter ’1’ / ’0’ to toggle induction on / off
first			First segment for which to compute site energies
last			Last segment for which to compute site energies
output			File to write site energies to. Site energies are also stored in the state file
check			Check mapping of polar sites to fragment
tholeparam			Thole parameters required for charge-smearing
cutoff		nm	Cut-off beyond which all interactions are neglected
cutoff2		nm	Cut-off beyond which polarization is neglected
expdamp			Damping exponent used in exponential damping function
scaling			1-n interaction scaling, currently not in use
esp			Control options for potential calculation
calcESP			Enter ’1’ / ’0’ to toggle on / off. If ’1’, site energies will not be evaluated
cube			
grid			XYZ file specifying grid points for potential evaluation
output			File to write grid-point potential to
esf			Control options for field calculation
calcESF			Enter ’1’ / ’0’ to toggle on / off. If ’1’, site energies will not be evaluated
grid			XYZ file specifying grid points for field evaluation
output			File to write grid-point field to
alphamol			Control options for molecular-polarizability calculation
calcAlpha			Enter ’1’ / ’0’ to toggle on / off. If ’1’, site energies will not be evaluated
output			File to write polarizability tensor in global frame and in diagonal form to
convparam			Convergence parameters for self-consistent field calculation
wSOR_N			Mixing factor for successive overrelaxation of neutral system, usually between 0.3 and 0.5
wSOR_C			Mixing factor for successive overrelaxation of charged system, usually between 0.3 and 0.5
tolerance			Convergence criterion, fulfilled if relative change smaller than tolerance
maxiter			Maximum number of iterations in the convergence loop

Return to the description of `zmultipole`.

edft

A wrapper for first principles based single site calculations

option	default	unit	description

job			Job options
tasks	input,run,pars		What to run
store	orbitals		What to store

Return to the description of `edft`.

idft

Projection method for electronic couplings. Requires edft otput

option	default	unit	description
tasks	input,run,pars		What to do
store	orbitals,overla		What to store
degeneracy	0	eV	Criterium for the degeneracy of two levels
levels	3		Output between HOMO, ..., HOMO-levels; LUMO, ..., LUMO+levels
trim	2		Use trim*occupied of virtual orbitals

Return to the description of `idft`.

pewald3d

Evaluates site energies in a periodic setting

option	default	unit	description
jobcontrol			
job_file			
multipoles			
mapping			
mps_table			
pdb_check			
coulombmethod			
method			
cutoff			
shape			
polarmethod			
method			
induce			
cutoff			
tasks			
calculate_fields			
polarize_fg			
evaluate_energy			
coarsegrain			
cg_background			
cg_foreground			
cg_radius			
cg_anisotropic			
convergence			
energy			
kfactor			
rfactor			

Return to the description of `pewald3d`.

qmmm

QM/MM with the Thole MM model

option	default	unit	description
control			
pdb_check			PDB file of polar sites
write_chk	dipoles.xyz		XYZ file with dipoles split onto point charges
format_chk	xyz		format, gaussian or xyz
split_dpl	1		’0’ do not split dipoles onto point charges, ’1’ do split
dpl_spacing	1e-3	nm	Spacing to be used when splitting dipole onto point charges: d = q * a
qmpackage			
package			QM package to use for the QM region
gwbse			Specify if GW/BSE excited state calculation ist needed
gwbse_options			GW/BSE options file
state			Number of excited state, which is to be calculated
type			Character of the excited state to be calculated
filter			Filter with which to find the excited state after each calculation
oscillator_strength			Oscillator strength filter, only states with higher oscillator strength are considered
charge_transfer			Charge transfer filter , only states with charge transfer above threshold are consdered
qmmmconvg			convergence criteria for the QM/MM
dR	0.001	nm	RMS of coordinates
dQ	0.001	e	RMS of charges
dE_QM	0.0001	eV	Energy change of the QM region
dE_MM	0.0001	eV	Energy change of the MM region
max_iter	10		Number of iterations
coulombmethod			Options for the MM embedding
method	cut-off		Method for evaluation of electrostatics
cutoff1			Cut-off for the polarizable MM1 shell
cutoff2			Cut-off for the static MM2 shell
tholemodel			Parameters for teh Thole model
induce			’1’ - induce ’0’ - no induction
induce_intra_pair			’1’ - include mutual interaction of induced dipoles in the QM region. ’0’ - do not
exp_damp	0.39		Sharpness parameter
scaling			Bond scaling factors
convergence			Convergence parameters for the MM1 (polarizable) region
wSOR_N			Mixing factor for the succesive overrelaxation algorithm for a neutral QM region
wSOR_C			Mixing factor for the succesive overrelaxation algorithm for a charged QM region
max_iter	512		Maximal number of iterations to converge induced dipoles
tolerance			Maximum RMS change allowed in induced dipoles

Return to the description of `qmmm`.

xqmultipole

Electrostatic interaction and induction energy of charged molecular clusters

option	default	unit	description

multipoles			Polar-site mapping definition
control			
job_file			Job file
emp_file			Polar-background definition, allocation of mps-files to segments
pdb_check			Whether or not to output a pdb-file of the mapped polar sites
format_chk			Format for check-file: ′xyz′ or ′gaussian′
split_dpl			Split dipoles onto point charges in check-file
dpl_spacing		nm	Spacing between point charges for check-file output
coulombmethod			
method			Currently only cut-off supported
cutoff1		nm	Full-interaction radius cut-off
cutoff2		nm	Radius of electrostatic buffer
tholemodel			
induce			Induce - or not
induce_intra_pair			Induce mutually within the charged cluster
exp_damp			Thole sharpness parameter
scaling			Bond scaling parameters, currently not used
convergence			
wSOR_N			SOR mixing factor for overall neutral clusters
wSOR_C			SOR mixing factor for overall charged clusters
max_iter			Maximum number of iterations
tolerance			Relative tolerance as convergence criterion

Return to the description of `xqmultipole`.

energy2xml

Write out energies from SQL file

option	default	unit	description

Return to the description of `energy2xml`.

integrals2xml

Write out transfer integrals from SQL file

option	default	unit	description

Return to the description of `integrals2xml`.

occupations2xml

Write out site occupation probabilities from SQL file

option	default	unit	description

Return to the description of `occupations2xml`.

pairs2xml

Write out neighbourlist from SQL file

option	default	unit	description

Return to the description of `pairs2xml`.

rates2xml

Write out charge transfer rates from SQL file

option	default	unit	description

Return to the description of `rates2xml`.

segments2xml

Write out segment data from SQL file

option	default	unit	description

Return to the description of `segments2xml`.

trajectory2pdb

Generate PDB files for the mapped MD/QM topology

option	default	unit	description

Return to the description of `trajectory2pdb`.

kmcmultiple

Kinetic Monte Carlo simulations of multiple holes or electrons in periodic boundary conditions

option	default	unit	description
runtime		seconds or integer	Simulated time in seconds (if a number smaller than 100 is given) or number of KMC steps (if a number larger than 100 is given)
outputtime	1E-8	seconds	Time difference between outputs into the trajectory file. Set to 0 if you wish to have no trajectory written out.
trajectoryfile	trajectory.csv		Name of the trajectory file
seed	123	integer	Integer to initialise the random number generator
injection	*		Name pattern that specifies on which sites injection is possible. Before injecting on a site it is checked whether the column 'name' in the table 'segments' of the state file matches this pattern. Use the wildcard '*' to inject on any site.
injectionmethod	random		Options: random/equilibrated. random: injection sites are selected randomly (generally the recommended option); equilibrated: sites are chosen such that the expected energy per carrier is matched, possibly speeding up convergence
numberofcharges	1	integer	Number of electrons/holes in the simulation box
fieldX	0	V/m	x component of the external electric field
fieldY	0	V/m	y component of the external electric field
fieldZ	1E6	V/m	z component of the external electric field
carriertype	electron		Options: electron/hole. Specifies the carrier type of the transport under consideration.
temperature		Kelvin	Temperature in Kelvin. Will only be relevant if rates are calculated by KMC and not taken from the state file.

explicitcoulomb	0		Options: 0/1/2. 0: no explit Coulomb interaction; 1: explicit Coulomb interaction using partial charges from SQL file, 2: explicit ’raw’ Coulomb interaction using charges of +/-1. Note that rates from the state file will not be used if Coulomb interaction is switched on (option 1 or 2) but rates will be calculated within KMC.
rates	statefile		Options: statefile/calculate. statefile: use the rates for charge transfer specified in the state file; calculate: use transfer integrals, site energies and reorganisation energies specified in the state file as well as temperature and electric field specified here to calculate rates before starting the KMC simulation. In case of explicit Coulomb interaction this option is set to ’calculate’ automatically. If you use rates from the state file make sure that the electric field specified here matches the one that was used for calculating the rates in the state file.

Return to the description of `kmcmultiple`.

pbc

Kinetic Monte Carlo simulation for multiple carriers in PBC

option	default	unit	description
runtime	0	seconds	Simulated time
nsteps	1	steps	Number of simulation steps
outtime	1E-8	seconds	Time difference between outputs into the trajectory file.
trajectoryfile	trajectory.csv		Name of the trajectory file
seed	123	integer	Integer to initialise the random number generator for events
injectionmethod	random		Random or uniform injection of the carriers.
nelectrons	1	integer	Number of electrons in the simulation box
nholes	0	integer	Number of holes in the simulation box
fieldX	0	V/m	x component of the external electric field
fieldY	0	V/m	y component of the external electric field
fieldZ	1E6	V/m	z component of the external electric field
temperature	300	K	Temperature of the system for rate calculation
rates	read		Method of rate input: read (from file) or calculate (in KMC)
interaction	Pauli		Method of interaction between like charge carriers (Pauli or Coulomb)

Return to the description of `pbc`.

Bibliography

[1] C. Poelking and D. Andrienko, Journal of Chemical Theory and Computation **12**, 4516 (2016). i, 14

[2] P. Kordt and D. Andrienko, J. Chem. Theory Comput. **12**, 36 (2016). i

[3] V. Rühle *et al.*, Journal of Chemical Theory and Computation **7**, 3335 (2011). i, 23, 24

[4] A. Lukyanov and D. Andrienko, Physical Review B **82**, 193202 (2010). i, 26

[5] B. Baumeier, J. Kirkpatrick, and D. Andrienko, Physical Chemistry Chemical Physics **12**, 11103 (2010). i, 19, 21, 23

[6] V. Rühle *et al.*, Journal of Chemical Theory and Computation **5**, 3211 (2009). i, 3

[7] J. Kirkpatrick, International Journal of Quantum Chemistry **108**, 51 (2008). i, 23

[8] V. Rühle, J. Kirkpatrick, and D. Andrienko, The Journal of Chemical Physics **132**, 134103 (2010). 3

[9] N. Vukmirović and L.-W. Wang, The Journal of Chemical Physics **128**, 121102 (2008). 4

[10] N. Vukmirović and L.-W. Wang, Nano Letters **9**, 3996 (2009).

[11] D. P. McMahon and A. Troisi, Chemical Physics Letters **480**, 210 (2009). 4

[12] J.-L. Brédas, D. Beljonne, V. Coropceanu, and J. Cornil, Chemical Reviews **104**, 4971 (2004). 8, 23

[13] V. May and O. Kühn, *Charge and Energy Transfer Dynamics in Molecular Systems*, 3rd, revised and enlarged edition ed. (Wiley-VCH, Weinheim, 2011). 9

[14] C. M. Breneman and K. B. Wiberg, Journal of Computational Chemistry **11**, 361 (1990). 11

[15] A. Stone and M. Alderton, Molecular Physics **56**, 1047 (1985). 11, 12

[16] L. E. Chirlian and M. M. Francl, Journal of Computational Chemistry **8**, 894 (1987). 11

[17] U. C. Singh and P. A. Kollman, Journal of Computational Chemistry **5**, 129 (1984). 12

[18] A. J. Stone, Journal of Chemical Theory and Computation **1**, 1128 (2005). 12

[19] A. J. Stone, *The Theory of intermolecular forces* (Clarendon Press, Oxford, 1997). 13

[20] B. T. Thole, Chemical Physics **59**, 341 (1981). 13

[21] P. T. van Duijnen and M. Swart, The Journal of Physical Chemistry A **102**, 2399 (1998). 13, 14

[22] J. Applequist, J. R. Carl, and K.-K. Fung, Journal of the American Chemical Society **94**, 2952 (1972). 13

[23] P. Ren and J. W. Ponder, The Journal of Physical Chemistry B **107**, 5933 (2003). 13, 14

[24] H. Baessler, Physica Status Solidi B **175**, 15 (1993). 19, 23

[25] A. Troisi and G. Orlandi, Physical Review Letters **96**, 086601 (2006).

[26] A. Troisi, D. L. Cheung, and D. Andrienko, Physical Review Letters **102**, 116602 (2009).

[27] D. P. McMahon and A. Troisi, ChemPhysChem **11**, 2067 (2010).

[28] T. Vehoff, B. Baumeier, A. Troisi, and D. Andrienko, Journal of the American Chemical Society **132**, 11702 (2010). 19

[29] A. B. Walker, A. Kambili, and S. J. Martin, Journal of Physics: Condensed Matter **14**, 9825 (2002). 23

[30] P. M. Borsenberger, L. Pautmeier, and H. Bässler, The Journal of Chemical Physics **94**, 5447 (1991).

[31] W. F. Pasveer *et al.*, Physical Review Letters **94**, 206601 (2005). 23

[32] J.-L. Brédas, J. E. Norton, J. Cornil, and V. Coropceanu, Accounts of Chemical Research **42**, 1691 (2009). 23

[33] V. Coropceanu *et al.*, Chemical Reviews **107**, 926 (2007).

[34] J. Nelson, J. J. Kwiatkowski, J. Kirkpatrick, and J. M. Frost, Accounts of Chemical Research **42**, 1768 (2009). 23

[35] R. A. Marcus, Reviews of Modern Physics **65**, 599 (1993). 24

[36] G. R. Hutchison, M. A. Ratner, and T. J. Marks, Journal of the American Chemical Society **127**, 2339 (2005). 24

[37] J.-L. Chang, Journal of Molecular Spectroscopy **232**, 102 (2005). 24

[38] B. M. Hoffman and M. A. Ratner, Inorganica Chimica Acta **243**, 233 (1996). 24

[39] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, Journal of Computational Physics **17**, 10 (1975). 25

[40] H. Scher and E. Montroll, Physical Review B **12**, 2455 (1975). 25

[41] P. M. Borsenberger, E. H. Magin, M. D. VanAuweraer, and F. C. D. Schryver, Physica Status Solidi A **140**, 9 (1993). 25

[42] B. Derrida, Journal of Statistical Physics **31**, 433 (1983). 26, 28

[43] H. Cordes *et al.*, Physical Review B **63**, 094201 (2001).

[44] K. Seki and M. Tachiya, Physical Review B **65**, 014305 (2001). 26

[45] J. J. M. van der Holst *et al.*, Physical Review B **79**, 085203 (2009). 27

[46] D. Dunlap, P. Parris, and V. Kenkre, Physical Review Letters **77**, 542 (1996). 28

[47] S. V. Novikov *et al.*, Physical Review Letters **81**, 4472 (1998). 28

[48] Y. Nagata and C. Lennartz, The Journal of Chemical Physics **129**, 034709 (2008). 28

[49] S. V. Novikov and A. V. Vannikov, The Journal of Physical Chemistry **99**, 14573 (1995). 28