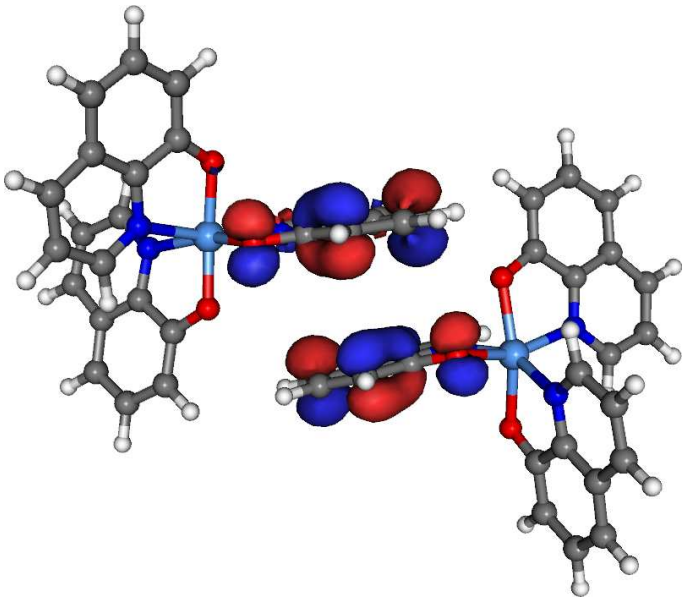
# VOTCA-XTP

## Exciton Transport Simulations

User Manual



compiled from: 1.4_rc1

September 26, 2016

www.votca.org

## Disclamer

This manual is not complete. The best way to start using the software is to look at provided tutorials. The reference section is generated automatically from the source code, so please make sure that your software and manual versions match.

## Citations

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

[1] 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

[2] 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

## Development

The core development is currently taking place at the Max Planck Institute for Polymer Research, Mainz, Germany and TU/e Eindhoven.

## Copyright

# Contents

# Index

# Chapter 1

# Introduction

sec:introduction

Charge carrier dynamics in an organic semiconductor can often be described in terms of charge hopping between localized states. The hopping rates depend on electronic coupling elements, reorganization energies, and site energies, which vary as a function of position and orientation of the molecules. The purpose of the VOTCA-XTP package [1] is to simplify the workflow for charge transport simulations, provide a uniform error-control for the methods, flexible platform for their development, and eventually allow *in silico* prescreening of organic semiconductors for specific applications.

The toolkit is implemented using modular concepts introduced earlier in the Versatile Object-oriented Toolkit for Coarse-graining Applications (VOTCA) [2]. It contains different programs, which execute specific tasks implemented in calculators representing an individual step in the workflow. Figure 1.1 summarizes a typical chain of commands to perform a charge transport simulation: First, the VOTCA code structures are adapted to reading atomistic trajectories, mapping them onto conjugated segments and rigid fragments, and substituting (if needed) rigid fragments with the optimized copies (`xtp_map`). The programs `xtp_run` and `xtp_parallel` (for heavy-duty tasks) are then used to calculate all bimolecular charge hopping rates (via precalculation of all required ingredients). Site energies (or energetic disorder) can be determined as a combination of internal (ionization potentials/electron affinities of single molecules) as well as electrostatic and polarization contributions within the molecular environment. The calculation of electronic coupling elements between conjugated segments from the corresponding molecular orbitals can be performed using a dimer-projection technique based on density-functional theory (DFT). This requires explicit calculations using quantum-chemistry software for which we provide interfaces to `Gaussian`, `Turbomole`, and `NWChem`. Alternatively, the molecular orbital overlap module calculates electronic coupling elements relying on the semi-empirical INDO Hamiltonian and molecular orbitals in the format provided by the `Gaussian` package.

The kinetic Monte Carlo module reads in the neighbor list, site coordinates, and hopping rates and performs charge dynamics simulations using either periodic boundary conditions or charge sources and sinks.

The toolkit is written as a combination of modular C++ code and scripts. The data transfer between programs is implemented via a state file (sql database), which is also used to restart simulations. Analysis functions and most of the calculation routines are encapsulated by using the observer pattern [3] which allows the implementation of new functions as individual modules.

In the following chapter 2, we summarize the theoretical background of the workflow of charge transport simulations and in particular its individual steps. Chapter 3 describes the structure and content of input and output files, while a full reference of programs and calculators is available in chapter 4. For a hands-on tutorial, the reader is referred to the VOTCA-XTP project page at http://code.google.com/p/votca-xtp/.

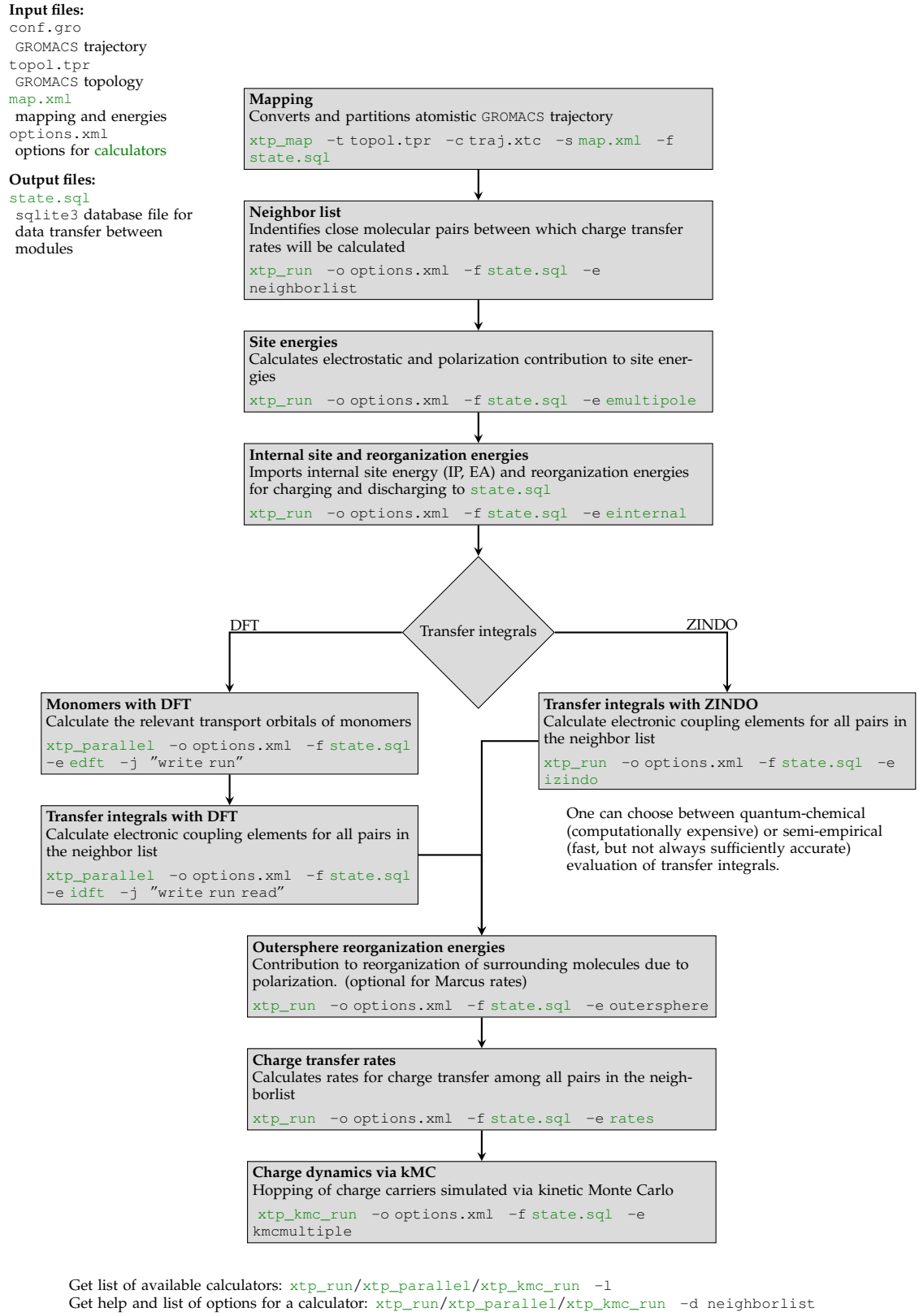



Figure 1.1: A practical workflow of charge transport simulations using VOTCA-XTP. The theoretical background of the individual steps is given in chapter 2. Chapter 3 describes the content of input and output files, while a full reference of programs and calculators is available in chapter 4. fig:summary

# Chapter 2

# Theoretical background

sec:theory

## 2.1 Workflow

sec:wokflow

A typical workflow of charge transport simulations is depicted in figure 2.1. The first step is the simulation of an atomistic morphology, which is then partitioned on hopping sites. The coordinates of the hopping sites are used to construct a list of pairs of molecules, or neighbor list.

fig:workflow

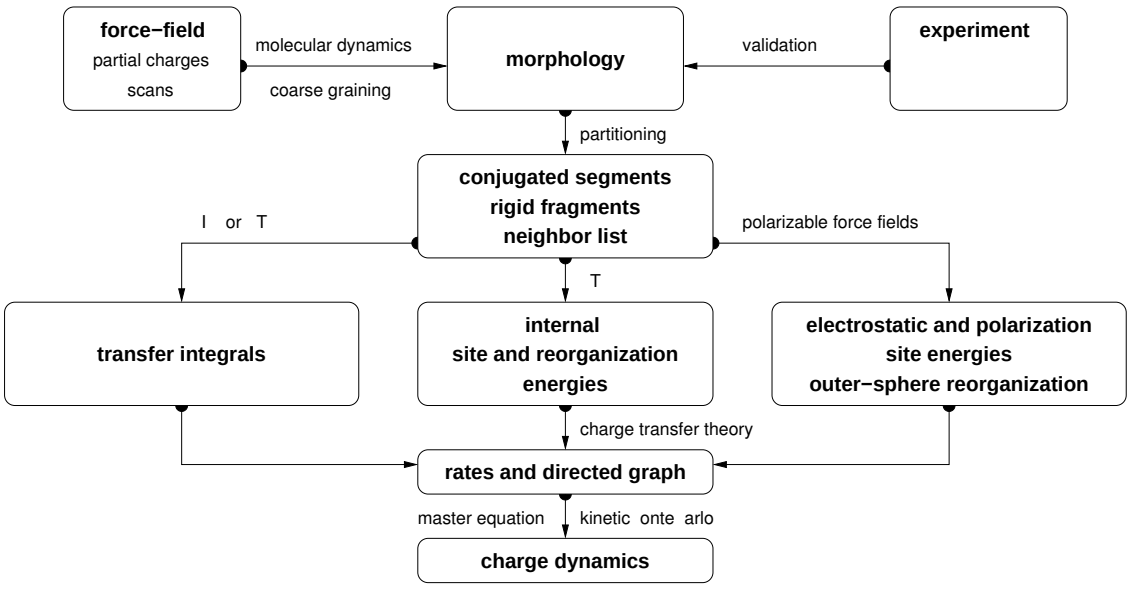



Figure 2.1: Workflow for microscopic simulations of charge transport.

For each pair an electronic coupling element, a reorganization energy, a driving force, and eventually the hopping rate are evaluated. The neighbor list and hopping rates define a directed graph. The corresponding master equation is solved using the kinetic Monte Carlo method, which allows to explicitly monitor the charge dynamics in the system as well as to calculate time or ensemble averages of occupation probabilities, charge fluxes, correlation functions, and field-dependent mobilities.

## 2.2 Material morphology

sec: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

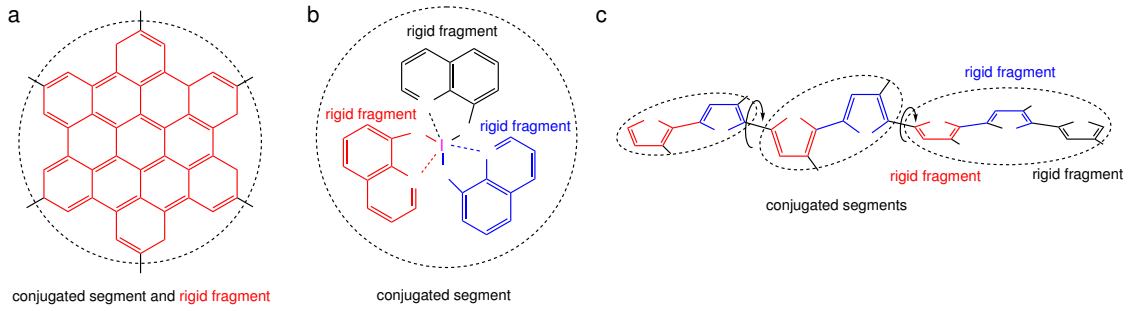



Figure 2.2: The concept of conjugated segments and rigid fragments. Dashed lines indicate conjugated segments while colors denote rigid fragments. (a) Hexabenzocoronene: the $\pi$-conjugated system is both a rigid fragment and a conjugated segment. (b) $Alq_3$: the Al atom and each ligand are rigid fragments 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. fig:segment

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 $\pi$-conjugated system. First-principles methods can be used to characterize the missing terms of the potential energy function.

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 [2]. 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 morphology is already known, that is we know how the topology and the coordinates of all atoms in the systems at a given time. VOTCA-XTP can read standard `GROMACS` topology files. Custom definitions of atomistic topology via XML files are also possible. Since the description of the atomistic topology is the first step in the charge transport simulations, it is important to follow simple conventions on how the system is partitioned on molecules, residues, and how atoms are named in the topology. Required input files are described in section atomistic topology.

## 2.3 Conjugated segments and rigid fragments

sec:segments

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 $\pi$-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 2.2. 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 $\pi$-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 $k_B T$ 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 [4]. However, such intuitive partitioning is, to some extent, arbitrary and shall be validated by other methods [5–7].

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 $\pi$-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.

To partition the system on hopping sites and substitute rigid fragments with the corresponding ground-state geometries `xtp_map` program is used:

**Mapping the GROMACS trajectory**

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

It reads in the `GROMACS` topology (`topol.tpr`) and trajectory (`traj.xtc`) files, definitions of conjugated segments and rigid fragments (`map.xml`) and outputs coordinates of conjugated segments (hopping sites) and rigid fragments (as provided in the MD trajectory and after rigidification) to the state file (`state.sql`). In order to do this, a mapping file `map.xml` has to be provided, which specifies the corresponding atoms in the different representations. After this step, all information (frame number, dimensions of the simulation box, etc) are stored in the state file and only this file is used for further calculations.

**Be careful!**

VOTCA-XTP requires a wrapped trajectory for mapping the segments and fragments, so all molecules should be whole in the frame.

In order to visually check the mapping one can use either the `tdump` calculator or the programm `xtp_dump` with the calculator `trajectory2pdb`.

sec:xtp_dump

**Writing a mapped trajectroy with `xtp_dump`**

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

It reads in the state file created by `xtp_map` and outputs two trajectory files corresponding to the original and rigidified atom coordinates. To check the mapping, it is useful to superimpose the three outputs (original atomistic, atomistic stored in the state file, and rigidified according to ground state geometries), e.g., with `VMD`.

sec:tdump

**Writing a mapped trajectroy with `tdump`**

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

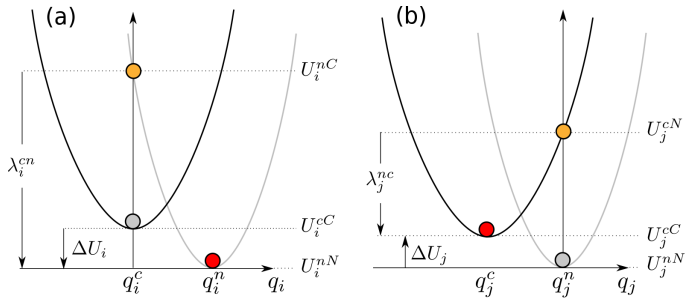



Figure 2.3: Potential energy surfaces of (a) donor and (b) acceptor in charged and neutral states. After the change of the charge state both molecules relax their nuclear coordinates. If all vibrational modes are treated classically, the total internal reorganization energy and the internal energy difference 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. fig:parabolas

It also reads in the state file but appends the coordinates to a pdb. file. So make sure to delete old QM.pdb and MD.pdb if you want to create a new imagef

## 2.4 Neighbor list

sec:neighborlist

A list of neigboring 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**

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

## 2.5 Reorganization energy

sec:reorganization

The reorganization energy $\lambda_{ij}$ 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.

### 2.5.1 Intramolecular reorganization energy

sec:eintramolecular

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 2.3.

Adding the contributions due to discharging of molecule $i$ and charging of molecule $j$ yields [8]

$$\lambda_{ij}^{\text{int}} = \lambda_i^{cn} + \lambda_j^{nc} = U_i^{nC} - U_i^{nN} + U_j^{cN} - U_j^{cC} \,. \tag{2.1}$$

equ:lambdas

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 ($\lambda^{cn}$) and charging ($\lambda^{nc}$) 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**

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

### 2.5.2 Outersphere reorganization energy

sec:eoutersphere

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 [9]

$$\lambda_{ij}^{\text{out}} = \frac{c_p}{2\epsilon_0} \int_{V^{\text{out}}} dV \left[ \vec{D}_I(\vec{r}) - \vec{D}_F(\vec{r}) \right]^2 \,, \tag{2.2}$$

equ:lambda_outer1

where $\epsilon_0$ 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, $V^{\text{out}}$ 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 ($\epsilon_{\text{s}}$) and high ($\epsilon_{\text{opt}}$) frequency dielectric permittivities.
Eq. (2.2) can be simplified by assuming spherically symmetric charge distributions on molecules $i$ and $j$ with total charge $e$. Integration over the volume $V^{\text{out}}$ 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

$$\lambda_{ij}^{\text{out}} = \frac{c_p e^2}{4\pi\epsilon_0} \left( \frac{1}{2R_i} + \frac{1}{2R_j} - \frac{1}{r_{ij}} \right) \,, \tag{2.3}$$

equ:lambda_outer2

where $r_{ij}$ is the molecular separation. While eq. (2.3) captures the main physics, e.g. predicts smaller outer-sphere 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

$$\vec{D}_I(\vec{r}_{b_k}) - \vec{D}_F(\vec{r}_{b_k}) = \sum_{a_i} \frac{q_{a_i}^c - q_{a_i}^n}{4\pi} \frac{(\vec{r}_{b_k} - \vec{r}_{a_i})}{|\vec{r}_{b_k} - \vec{r}_{a_i}|^3} + \sum_{a_j} \frac{q_{a_j}^n - q_{a_j}^c}{4\pi} \frac{(\vec{r}_{b_k} - \vec{r}_{a_j})}{|\vec{r}_{b_k} - \vec{r}_{a_j}|^3} \,, \tag{2.4}$$

where $q_{a_i}^n$ ($q_{a_i}^c$) 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. (2.2) 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 $\epsilon_{\text{opt}}$. 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. (2.2). 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. (2.3), 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**

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

## 2.6 Site energies

sec: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, $\omega_{ij}$, depends exponentially on $\Delta E_{ij}$ (see eq. (2.31)) 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.

### 2.6.1 Externally applied electric field

sec:ext_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 $1\,\text{nm}$, and moderate fields of $F < 10^8\,\text{V/m}$ this term is always smaller than $0.1\,\text{eV}$.

### 2.6.2 Internal energy

sec:internal_energy

The contribution to the site energy difference due to different internal energies (see figure 2.3) can be written as

$$\Delta E_{ij}^{\text{int}} = \Delta U_i - \Delta U_j = \left(U_i^{cC} - U_i^{nN}\right) - \left(U_j^{cC} - U_j^{nN}\right) , \tag{2.5}$$

equ:conformational

where $U_i^{cC(nN)}$ is the total energy of molecule $i$ in the charged (neutral) state and geometry. $\Delta U_i$ corresponds to the adiabatic ionization potential (or electron affinity) of molecule $i$, as shown in figure 2.3. 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**

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

### 2.6.3 Electrostatic interaction energy

sec:distributed_multipoles

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 $b\epsilon B$,

$$U_{AB} = \sum_{a\epsilon A}\sum_{b\epsilon B} \hat{Q}^a_{l_1k_1} T^{a,b}_{l_1k_1l_2k_2} \hat{Q}^b_{l_2k_2} \equiv \hat{Q}^a_{l_1k_1} T^{a,b}_{l_1k_1l_2k_2} \hat{Q}^b_{l_2k_2}, \tag{2.6}$$

equ:mol_distributed_U

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 [10] from category (a) and DMA [11] 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 [10]. The fitted charges result from minimizing the Lagrangian function [12]

$$z(\{q_i\}) = \sum_{k=1}^{M}\left(\phi(\vec{r}_k) - \sum_{i=1}^{N}\frac{1}{4\pi\varepsilon_0}\frac{q_i}{|\vec{r}_i - \vec{r}_k|}\right) + \lambda\left(q_{\text{mol}} - \sum_{i=1}^{N} q_i\right), \tag{2.7}$$

with $M$ grid points, $N$ atomic sites, the set of atomic partial charges $\{q_i\}$ and the SCF potential $\phi$. The Lagrange multiplier $\lambda$ constrains the sum of the fitted charges to the molecular charge $q_{\text{mol}}$. The main difference from other fitting schemes [13] 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 [11, 14], developed by A. Stone, operates directly on the quantum-mechanical density matrix, expanded in terms of atom- and bond-centered Gaussian functions $\chi_\alpha = R_{LK}(\vec{x} - \vec{s}_\alpha)\exp[-\zeta(\vec{x} - \vec{s}_\alpha)^2]$,

$$\rho(\vec{x}) = \sum_{\alpha,\beta} \rho_{\alpha\beta}\chi_\alpha(\vec{x} - \vec{s}_\alpha)\chi_\beta(\vec{x} - \vec{s}_\beta). \tag{2.8}$$

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 $\vec{P}$:

$$Q_{LK}[\vec{P}] = -\int R_{LK}(\vec{x}-\vec{P})\rho_{\alpha\beta}\chi_\alpha\chi_\beta d^3x. \tag{2.9}$$

Second, we transfer the resulting $Q_{lk}[\vec{P}]$ to the position $\vec{S}$ of a polar site according to the rule [11]

$$Q_{nm}[\vec{S}] = \sum_{l=0}^{L}\sum_{k=-l}^{l}\left[\begin{pmatrix} n+m \\ l+k \end{pmatrix}\begin{pmatrix} n-m \\ l-k \end{pmatrix}\right]^{1/2} R_{n-l,m-k}(\vec{S}-\vec{P})\cdot Q_{lk}[\vec{P}]. \tag{2.10}$$

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 [14], 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 sxtpth 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 [14] (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**

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

### 2.6.4 Induction energy - the Thole model

sec:thole_model

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),

$$U_{\text{int}} = \frac{1}{2}\sum_A \Delta Q_t^a \eta_{tt'}^{aa'} \Delta Q_{t'}^{a'}, \tag{2.11}$$

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 $1/2$ [15]:

$$U_{pu} = \frac{1}{2}\sum_A\sum_{B>A}\left[\Delta Q_t^a T_{tu}^{ab} Q_u^b + \Delta Q_t^b T_{tu}^{ab} Q_u^a\right]. \tag{2.12}$$ equ:u_pu

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

$$\Delta Q_t^a = -\sum_{B\neq A}\alpha_{tt'}^{aa'} T_{t'u}^{a'b}(Q_u^b + \Delta Q_u^b), \tag{2.13}$$ equ:self_consistent_dQ

where the polarizability tensors $\alpha_{tt'}^{aa'}$ are given by the inverse of $\eta_{tt'}^{aa'}$.
With eqs. 2.13 and 2.12 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 [16, 17] 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 [16–18]. 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 $\alpha^a$. This isotropic fractional charge density gives rise to a modified potential

$$\phi(u) = -\frac{1}{4\pi\varepsilon_0}\int_0^u 4\pi u'\rho(u')du' \tag{2.14}$$ equ:mod_potential

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 $\vec{u}$,

$$T_{ij\ldots}(\vec{R}) = f(\alpha^a\alpha^b)\, t_{ij\ldots}(\vec{u}(\vec{R}, \alpha^a\alpha^b)), \tag{2.15}$$

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. 2.14,

$$t_{ij\ldots}(\vec{u}) = -\partial_{u_i}\partial_{u_j}\ldots\phi(\vec{u}). \tag{2.16}$$

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 $\lambda_n(\vec{u}(\vec{R}))$ [19].
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 [17]. 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

$$\rho(u) = \frac{3a}{4\pi}\exp(-au^3), \tag{2.17}$$

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 [19].
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**

```
xtp_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**

```
xtp_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**

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

## 2.7 Transfer integrals

sec:transfer_integrals

The electronic transfer integral element $J_{ij}$ entering the Marcus rates in eq. (2.31) is defined as

$$J_{ij} = \left\langle \phi_i \left| \hat{H} \right| \phi_j \right\rangle, \tag{2.18}$$

equ:TI

where $\phi_i$ and $\phi_j$ 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 $\phi_i$ 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 [20–24], 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. (2.18) 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.

### 2.7.1 Projection of monomer orbitals on dimer orbitals (DIPRO)

sec:dipro

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. [25]. A brief summary of the concept is given below.

We start from an effective Hamiltonian [1]

$$\hat{H}^{\text{eff}} = \sum_i \epsilon_i \hat{a}_i^\dagger \hat{a}_i + \sum_{j \neq i} J_{ij} \hat{a}_i^\dagger \hat{a}_j + c.c. \tag{2.19}$$ equ:dipro_eq1

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 $\epsilon_i$, 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) $\phi_i$ and $\phi_j$, 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 $\phi_i$ and $\phi_j$. The expansion coefficients, $\bar{\mathbf{C}}$, can be determined by solving the secular equation

$$(\mathbf{H} - E\mathbf{S})\bar{\mathbf{C}} = 0 \tag{2.20}$$ equ:dipro_eq2

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

$$\mathbf{H} = \begin{pmatrix} e_i & H_{ij} \\ H_{ij}^* & e_j \end{pmatrix} \qquad \mathbf{S} = \begin{pmatrix} 1 & S_{ij} \\ S_{ij}^* & 1 \end{pmatrix} \tag{2.21}$$ equ:dipro_eq3

with

$$\begin{aligned} e_i &= \langle \phi_i | \hat{H} | \phi_i \rangle & H_{ij} &= \langle \phi_i | \hat{H} | \phi_j \rangle \\ e_j &= \langle \phi_j | \hat{H} | \phi_j \rangle & S_{ij} &= \langle \phi_j | \phi_j \rangle \end{aligned} \tag{2.22}$$ equ:dipro_eq4

The matrix elements $e_{i(j)}$, $H_{ij}$, and $S_{ij}$ entering eq. (2.21) can be calculated via projections on the dimer orbitals (eigenfunctions of $\hat{H}$) $\{|\phi_n^{\text{D}}\rangle\}$ by inserting $\hat{1} = \sum_n |\phi_n^{\text{D}}\rangle \langle \phi_n^{\text{D}}|$ twice. We exemplify this explicitly for $H_{ij}$ in the following

$$H_{ij} = \sum_{nm} \langle \phi_i \,|\, \phi_n^{\text{D}} \rangle \langle \phi_n^{D} | \hat{H} | \phi_m^{\text{D}} \rangle \langle \phi_m^{\text{D}} \,|\, \phi_j \rangle. \tag{2.23}$$ eq:dipro_eq16

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

$$H_{ij} = \bar{\mathbf{V}}_{(i)} \mathbf{E} \bar{\mathbf{V}}_{(j)}^\dagger. \tag{2.24}$$ eq:dipro_eq17

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

$$|\phi_k\rangle = \sum_\alpha \lambda_\alpha^{(k)} |\varphi_\alpha\rangle \qquad \text{and} \qquad |\phi_n^{\text{D}}\rangle = \sum_\alpha D_\alpha^{(n)} |\varphi_\alpha\rangle \tag{2.25}$$ eq:dipro_eq18

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

$$\left(\bar{\mathbf{V}}_k\right)_n = \langle \phi_k \,|\, \phi_n^{\text{D}} \rangle = \sum_\alpha \lambda_\alpha^{(k)} \langle \alpha | \sum_\beta D_\beta^{(n)} |\beta\rangle = \bar{\boldsymbol{\lambda}}_{(k)}^\dagger \mathcal{S} \bar{\mathbf{D}}_{(n)} \tag{2.26}$$ eq:dipro_eq19

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

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. (2.21) as:

$$\begin{aligned} H_{ij} &= \bar{\boldsymbol{\lambda}}^{\dagger}_{(i)} \mathcal{S}\mathbf{D}\mathbf{E}\mathbf{D}^{\dagger}\mathcal{S}^{\dagger}\bar{\boldsymbol{\lambda}}_{(j)} \\ S_{ij} &= \bar{\boldsymbol{\lambda}}^{\dagger}_{(i)} \mathcal{S}\mathbf{D}\mathbf{D}^{\dagger}\mathcal{S}^{\dagger}\bar{\boldsymbol{\lambda}}_{(j)} \end{aligned} \tag{2.27}$$

eq:dipro_eq20

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. (2.20) into a standard eigenvalue problem of the form

$$\mathbf{H}^{\text{eff}}\bar{\mathbf{C}}^{\text{eff}} = E\bar{\mathbf{C}}^{\text{eff}} \tag{2.28}$$

eq:dipro_eq7

to make it correspond to eq. (2.19). According to Löwdin such a transformation can be achieved by

$$\mathbf{H}^{\text{eff}} = \mathbf{S}^{-1/2}\mathbf{H}\mathbf{S}^{-1/2}. \tag{2.29}$$

eq:dipro_eq9

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

$$\mathbf{H}^{\text{eff}} = \begin{pmatrix} e_i^{\text{eff}} & H_{ij}^{\text{eff}} \\ H_{ij}^{*,\text{eff}} & e_j^{\text{eff}} \end{pmatrix} = \begin{pmatrix} \epsilon_i & J_{ij} \\ J_{ij}^{*} & \epsilon_j \end{pmatrix} \tag{2.30}$$

eq:dipro_eq11

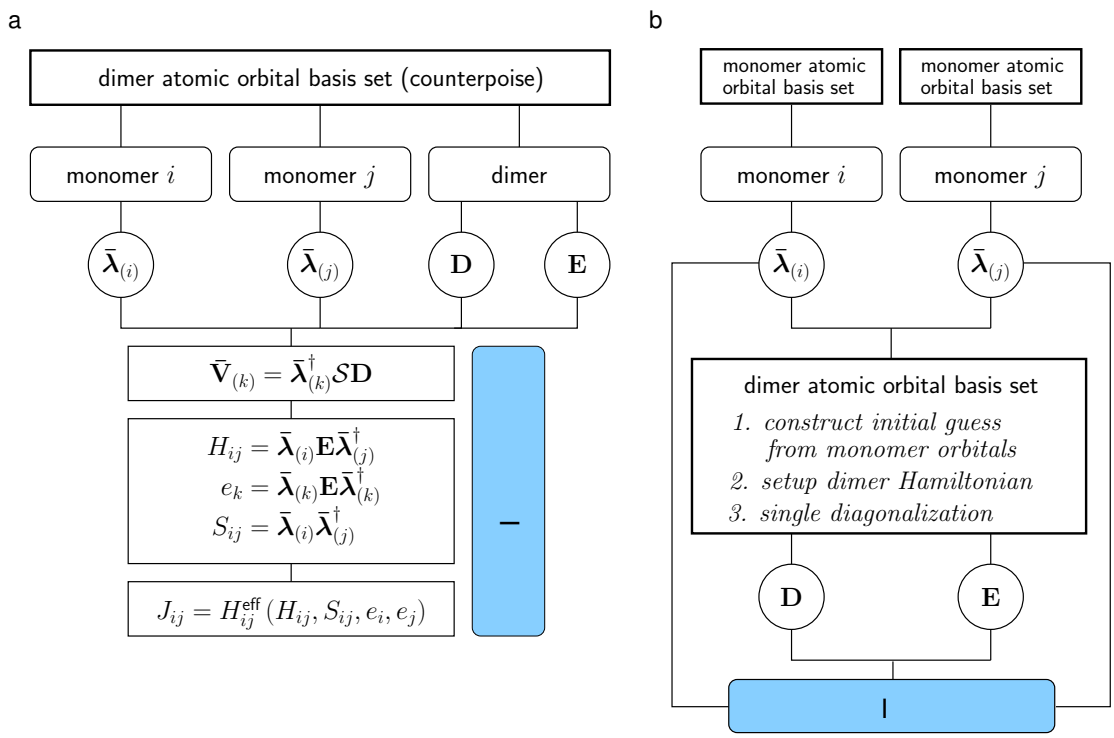



Figure 2.4: Schematics of the DIPRO method. (a) General workflow of the projection technique. (b) Strategy of the efficient noCP+noSCF implementation, in which the monomer calculations are performed independently form the dimer configurations (noCP), using the `edft` calculator. The dimer Hamiltonian 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.

fig:dipro_scheme

### 2.7.2 DFT-based transfer integrals using DIPRO

sec:dft

The calculation of one electronic coupling element based on DFT using the DIPRO method 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 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 2.4(a) and recommended for production runs. A detailed comparative study of the different variants can be found in [25].
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

sec:edft

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

2. Running of all jobs in job file

   **Running all edft jobs**
   ```
   xtp_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

sec:idft

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`**

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

2. Running of all jobs in job file

**Running all `idft` jobs**

```
xtp_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`**

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

### 2.7.3 ZINDO-based transfer integrals using MOO

sec:izindo

An approximate method based on Zerner's Intermediate Neglect of Differential Overlap (ZINDO) has been described in Ref. [26]. 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-XTP.

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.

**Be careful!**

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`**

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

## 2.8 Charge transfer rate

sec:rates

Charge transfer rates can be postulated based on intuitive physical considerations, as it is done in the Gaussian disorder models [20, 27–29]. Alternatively, charge transfer theories can be used to evaluate rates from quantum chemical calculations [1, 8, 25, 30–32]. 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.

### 2.8.1 Classical charge transfer rate

sec:rate_classical

The high temperature limit of classical charge transfer theory [33, 34] 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

$$\omega_{ij} = \frac{2\pi}{\hbar} \frac{J_{ij}^2}{\sqrt{4\pi\lambda_{ij}k_\mathrm{B}T}} \exp\left[-\frac{(\Delta E_{ij} - \lambda_{ij})^2}{4\lambda_{ij}k_\mathrm{B}T}\right], \tag{2.31}$$

equ:marcus

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, $\Delta E_{ij}$ is the site-energy difference, or driving force, and $J_{ij}$ is the electronic coupling element, or transfer integral.

### 2.8.2 Semi-classical bimolecular rate

sec:rate_bimolecular

The main assumptions in eq. (2.31) 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. [1] and has the following form

$$\omega_{ij} = \frac{2\pi}{\hbar} \frac{|J_{ij}|^2}{\sqrt{4\pi\lambda_{ij}^{\mathrm{out}}k_\mathrm{B}T}} \sum_{l',m'=0}^{\infty} |\langle\chi_{i0}^c|\chi_{il'}^n\rangle|^2 |\langle\chi_{j0}^n|\chi_{jm'}^c\rangle|^2 \exp\left\{-\frac{\left[\Delta E_{ij} - \hbar(l'\omega_i^n + m'\omega_j^c) - \lambda_{ij}^{\mathrm{out}}\right]^2}{4\lambda_{ij}^{\mathrm{out}}k_\mathrm{B}T}\right\}. \tag{2.32}$$

equ:jjortner

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 [35]

$$|\langle\chi_{i0}^c|\chi_{il'}^n\rangle|^2 = \frac{2}{2^{l'}l'!}\frac{\sqrt{\omega_i^c\omega_i^n}}{(\omega_i^c+\omega_i^n)}\exp\left(-|s_i|\right)\left[\sum_{\substack{k=0\\k\text{ even}}}^{l'}\binom{l'}{k}\left(\frac{2\omega_i^c}{\omega_i^c+\omega_i^n}\right)^{k/2}\frac{k!}{(k/2)!}H_{l'-k}\left(\frac{s_i}{\sqrt{2S_i^{cn}}}\right)\right]^2, \tag{2.33}$$

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 $\lambda_i^{cn}$ can be computed from the intramolecular PES.

### 2.8.3 Semi-classical rate

sec:rate_semiclassical

One can also use the quantum-classical rate with a common set of vibrational coordinates [9]

$$\omega_{ij} = \frac{2\pi}{\hbar}\frac{|J_{ij}|^2}{\sqrt{4\pi\lambda_{ij}^{\text{out}}k_{\text{B}}T}}\sum_{N=0}^{\infty}\frac{1}{N!}\left(\frac{\lambda_{ij}^{\text{int}}}{\hbar\omega^{\text{int}}}\right)^N\exp\left(-\frac{\lambda_{ij}^{\text{int}}}{\hbar\omega^{\text{int}}}\right)\exp\left\{-\frac{\left[\Delta E_{ij}-\hbar N\omega^{\text{int}}-\lambda_{ij}^{\text{out}}\right]^2}{4\lambda_{ij}^{\text{out}}k_{\text{B}}T}\right\}. \tag{2.34}$$

equ:jortner

Numerical estimates show that if $\lambda_{ij}^{\text{int}} \approx \lambda_{ij}^{\text{out}}$ and $|\Delta E_{ij}| \ll \lambda_{ij}^{\text{out}}$ the rates are similar to those of eq. (2.31). In general, there is no robust method to compute $\lambda_{ij}^{\text{out}}$ [36] 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**

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

## 2.9 Master equation

sec:kmc

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

$$\frac{\partial P_\alpha}{\partial t} = \sum_\beta P_\beta\Omega_{\beta\alpha} - \sum_\beta P_\alpha\Omega_{\alpha\beta}, \tag{2.35}$$

equ:master

where $P_\alpha$ is the probability of the system to be in a state $\alpha$ at time $t$ and $\Omega_{\alpha\beta}$ is the transition rate from state $\alpha$ to state $\beta$. A state $\alpha$ 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 $\omega_{ij}$.
The solution of eq. (2.35) 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 [37] and the first reaction method is implemented.
To obtain the dynamics of charges using KMC, the program `xtp_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**

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

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

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

### 2.9.1 Extrapolation to nondispersive mobilities

sec:nondispersive

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 [38, 39] 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 [40–42]. 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 [43].
For $\mathrm{Alq}_3$, 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 [43]. 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 $\sigma$, since $N$ increases exponentially with $\sigma^2$.

## 2.10 Stochastic Networks

sec:stochastic

The VOTCA package contains the functionality of generating large, amorphous charge transport networks ($\sim 10^6$ molecules). This is done with a combined coarse-grained and stochastic approach. VOTCA::CSG is used to generate a coarse-grained morphology. The stochastic modeling of VOTCA::CTP allows to make a charge transfer network out of this morphology by reproducing the neighbor list (connectivity), transfer integrals, correlated site energies. An overview is given in Figure 2.5.
Througout this section we will use two state files. One is the state file `state_ref.sql` of the smaller reference system that can be generated as explained in the previous sections. The second one is the state file `state_cg.sql` of the coarse-grained system, or the stochastic network, that can be parametrized as explained in this section.
When using the stochastic functionalities, please cite the corresponding work:

1. B. Baumeier, O. Stenzel, C. Poelking, D. Andrienko, and V. Schmidt: Stochastic modeling of molecular charge transport networks. Phys. Rev. B 86, 184202 (2012)

2. P. Kordt and D. Andrienko: Modeling of Spatially Correlated Energetic Disorder in Organic Semiconductors. Journal of Chemical Theory and Computation 12, 36–40 (2016)

3. P. Kordt, J. J. M. van der Holst, M. Al Helwi, W. Kowalsky, F. May, A. Badinski, C. Lennartz, and D. Andrienko: Modeling of Organic Light Emitting Diodes: From Molecular to Device Properties. Advanced Functional Materials 25, 1955–1971 (2015)

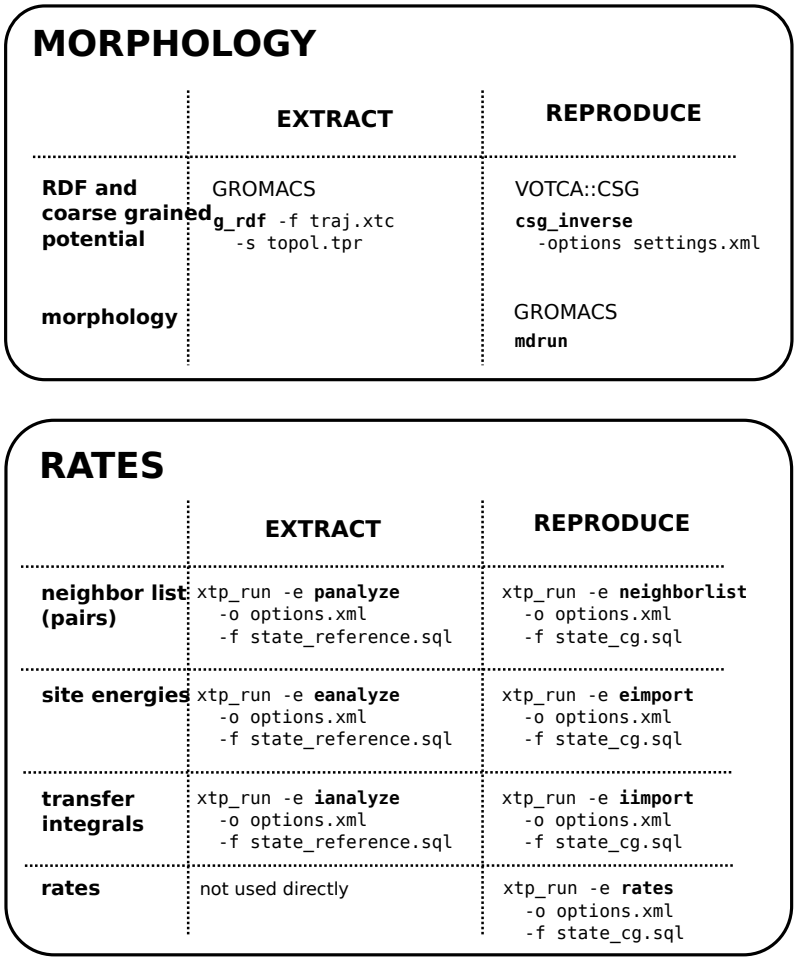



Figure 2.5: Stochastic Model in VOTCA. Overview of the different steps for generating stochastic charge transport networks in VOTCA. The Molecular Dynamics software GROMACS allows to analyze the radial distribution function of a morphology, which is then used by VOTCA::CSG to generate a coarse-grained potential that reproduces this distribution function. This potential can then be used for coarse-grained simulations in GROMCAS. For calculating rates in the coarse-grained morpholgy, first the relavant parameters are extracted (panalyze, eanalyze, ianalyze) from the reference morphology and and then reproduced in the coarsed-coarse grained morphology (neighborlist, eimport, iimport). With all these at hand, the rates calculator can be used in the coarse-grained morphology. fig:overview_stochastic

### 2.10.1 Coarse-grained morphology

The first step is to generate a coarse-grained morphology. In this example, it is done by mapping a DPBIC molecule (which consists of 103 atoms) to a single point, its center of mass and by using

the iterative Boltzmann inversion (IBI) method. Starting point is a smaller reference morphology, generated with GROMACS. Using the command

```
g_rdf -f traj.xtc -s topol.tpr
```

you can extract the radial distribution function $g(r)$ of your reference topolgy, outputted into the file rdf.xvg. This file, together with table.xvg,grompp.mdp, topol.top, index.ndx and confout.gro form your reference data.

For VOTCA::CSG you need a **setting.xml** file:

```
<cg>
 <!-- example for a non-bonded interaction entry -->
 <non-bonded>
  <!-- name of the interaction -->
  <name>IR-IR</name>
  <!-- types involved in this interaction -->
  <type1>IR</type1>
  <type2>IR</type2>
  <!-- dimension + grid spacing of tables for calculations -->
  <min>0.5</min>
  <max>5.0</max>
  <step>0.01</step>
  <inverse>
   <!-- target distribution (rdf), just give gromacs rdf.xvg -->
   <target>rdf.xvg</target>
   <!-- update cycles -->
   <do_potential>1</do_potential>
   <!-- additional post processing of dU before added to potential -->
   <post_update>scale smooth</post_update>
   <post_update_options>
          <scale>0.5</scale> <!--Scale the potential before updating it -->
        <smooth>
                <iterations>2</iterations>
        </smooth>
   </post_update_options>
   <!-- additional post processing of U after dU added to potential -->
   <post_add></post_add>
   <!-- name of the table for gromacs run -->
   <gromacs>
    <table>table_IR_IR.xvg</table>
   </gromacs>
  </inverse>
 </non-bonded>

 <!-- general options for inverse script -->
 <inverse>
  <!-- 300*0.00831451 gromacs units -->
  <kBT>2.49435300</kBT>
  <initial_configuration>maindir</initial_configuration>
  <!-- use gromacs as simulation program -->
  <program>gromacs</program>
  <!-- gromacs specific options -->
  <gromacs>
     <!-- trash so many frames at the beginning -->
     <equi_time>500</equi_time>
     <!-- grid for table*.xvg !-->
     <table_bins>0.001</table_bins>
     <!-- cut the potential at this value (gromacs bug) -->
     <pot_max>1000000</pot_max>
     <!-- extend the tables to this value -->
```

```
    <table_end>6.0</table_end>
  </gromacs>
  <!-- these files are copied for each new run -->
  <filelist>grompp.mdp topol.top index.ndx table.xvg</filelist>
  <!-- do so many iterations -->
  <iterations_max>500</iterations_max>
  <!-- Try to clean a bit -->
  <cleanlist>traj.xtc</cleanlist>
  <!-- ibm: inverse boltzmann imc: inverse monte carlo -->
  <method>ibi</method>
  <!-- write log to this file -->
  <log_file>inverse.log</log_file>
  <!-- write restart step to this file -->
  <restart_file>restart_points.log</restart_file>
  <!-- imc specific stuff -->
 </inverse>
</cg>
```

You run IBI using the command

```
csg_inverse -options settings.xml
```

IBI intents to find a pontential $U(r)$ that reproduces your radial distribution function. It is stored in the file **table_IR_IR.xvg** in our example.

With the interaction potential at hand, a large topology can be generated using molecular dynamics simulations for the coarse grained model. Starting point is a box with equally distributed points, with each point representing one molecule and with the number of points chosen such that the density of the reference system is reproduced. A small python script can generate the conf.gro to start from, here shown to obtain a $50 \times 50 \times 120\ \mathrm{nm}^3$ starting morphology.

```
from pylab import *
import numpy as np

lenX      =  50
lenY      =  50
lenZ      =  120
originalV = 4704.339
originalN = 4000
spacing   = (originalV/originalN)**(1./3.)

molecule  = "DPBIC"
resname   = "IRI"
atomname  = "IR"

newV = lenX*lenY*lenZ
newN = int(newV/originalV*originalN)

nX   = int(lenX/spacing)+1
nY   = int(lenY/spacing)+1
nZ   = int(lenZ/spacing)+1

print "max. molecules in X direction:  "+str(nX)
print "max. molecules in Y direction:  "+str(nY)
print "max. molecules in Z direction:  "+str(nZ)
print "total number of molecules: "+str(newN)

file = open("box.gro", "w")
file.write(molecule+"\n")
file.write(str(newN)+"\n")
```

```
atomnumber = 1
for iX in range(nX):
    for iY in range(nY):
        for iZ in range(nZ):
            if(atomnumber > newN):
                break
            posX = spacing*iX
            posY = spacing*iY
            posZ = spacing*iZ
            print >> file, "%5d%-5s%5s%5d%8.3f%8.3f%8.3f%8.4f%8.4f%8.4f" % \
            (1,resname,atomname,1,posX,posY,posZ,0,0,0)
            atomnumber += 1


file.write("  "+str(lenX)+"  "+str(lenY)+"  "+str(lenZ))
file.close()

print "Note: for some obscure reason VMD will not be able to read this file\
properly unless you open it once in vi and save it."
```

Open the box.gro in vi and save it (`:wq`), afterwards you can have a look at it in VDM. Run your MD simulations using the `mdrun` command. In the end you can compare the radial distribution functions of your reference and coarse-grained system, as shown in figure 2.6(a) as an example.

### 2.10.2 Charge transport network

To generate a charge transport network you first need a reference system with neighbor list, site energies and transfer integrals calculated and stored in a **state.sql** state file. The procedure for all these three properties is always the same: first analyze the reference data, and second import the analyzation files and reproduce the properties.

#### Neighbor list

In the atomistic reference system molecules are connected if their two closest segments are below a certain cut-off radius. This finer picture of segments does not exist in the coarse-grained system, where each molecule is represented by a point. To mimick the neighbor list, the probability of two molecules to be connected is analyzed as a function of their center-of-mass distance. This can be done by using the *panalyze* calculator

**Analyze the pair connectivity (neighborlist) in the reference system**

```
xtp_run -o options.xml -f state_ref.sql -e panalyze
```

with the options defined as follows:

**options_analyze.xml**

```
<options>
        <panalyze>
                <resolution_space>0.05</resolution_space>
        </panalyze>
</options>
```

The only parameter needed is the spacial resolution, i.e., the bin size for calculating the probabilities. The *panalyze* calculator outputs a file **panalyze.distanceprobability.out** with the respective probabilities. Now this file has to be imported into the coarse-grained state file

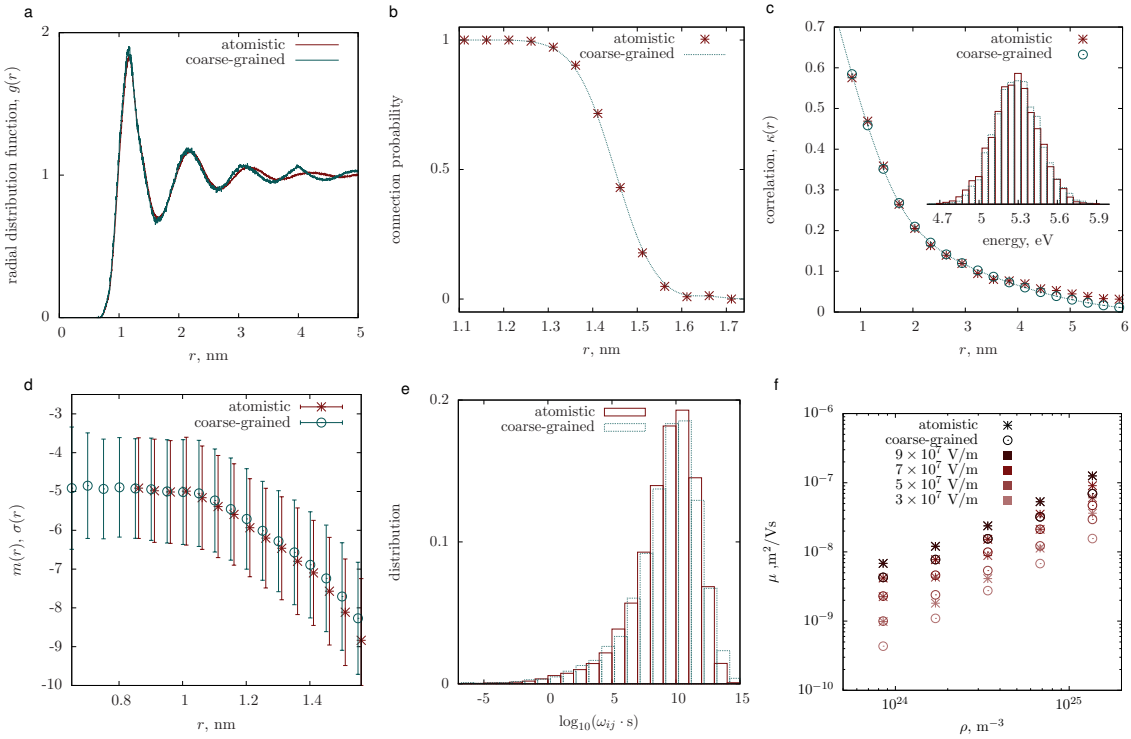



Figure 2.6: Comparison of the atomistic ($17 \times 17 \times 17\,\mathrm{nm}^3$) and coarse-grained ($50 \times 50 \times 120\,\mathrm{nm}^3$) models. (a) Radial distribution function, $g(r)$. (b) Probability of two sites to be connected (added to the neighbor list) as a function of their separation. (c) Spatial site energy autocorrelation function, $\kappa(r)$; Inset: Site energy distribution. (d) Mean $m$ and width $\sigma$ of a distribution of the logarithm of electronic couplings, $\log_{10}(J^2/\,\mathrm{eV}^2)$, for molecules at a fixed separation $r$. (e) Rate distributions. (f) Mobility as a function of hole density, plotted for four different electric fields. fig:stochastic

**Import the reference pair connectivity (neighborlist) and reproduce it in stochastic network**

```
xtp_run -o options.xml -f state_cg.sql -e neighborlist
```

using the following options:

**options_import.xml**

```
<options>
        <neighborlist>
                <probabilityfile>panalyze.distanceprobability.out</probabilityfile>
        </neighborlist>
</options>
```

For testing purposes, you can run the *panalyze* calculator on your coarse-grained state file and compare the probability function to the reference. An example is shown in figure 2.6(b). You can also also look at the file **panalyze.distanceprobability.out** for both state files, which has the distribution of coordination numbers (number of neighbors) and its average in.

### Site energies

Site energies in amorphous organic semiconductors are roughly Gaussian distributed, with the width of the Gaussian, $\sigma$, called the energetic disorder. However, there are correlations between sites if they are close enough to each other. The aim in this section is therefore to reproduce the

correlated energetic landscape. The first step is to get a spatial correlation function as well as the mean energy and the energetic disorder from your reference state file:

**Analyze the energy distribution and correlation in the reference system**

```
xtp_run -o options.xml -f state_ref.sql -e eanalyze
```

with the following options:

**options_analyze.xml**

```
<options>
        <eanalyze>
                <resolution_sites>0.05</resolution_sites>
                <resolution_pairs>0.05</resolution_pairs>
                <resolution_space>0.3</resolution_space>
                <states>1,-1</states> <!-- +1 for hole transport, -1 for electron transport --
                <distancemode>centreofmass</distancemode>
        </eanalyze>
</options>
```

The first three parameters determine bin sizes, then you can choose to look at hole and/or electron energy. The keyword *centreofmass* means, that the correlation function is calculated as a function of the centre-of-mass distance of molecules and not as a function of their nearest segments. For the stochastic simulations you always have to use the *centreofmass* mode!
The output files of this calculator that we need are **eanalyze.sitecorr_e.out** (for electrons) and **eanalyze.sitecorr_h.out** (for holes). In the second line of this file, you find mean and sigma of the energy distribution, as well as the mean of the static energies (without induction):

```
# EANALYZE: SPATIAL SITE-ENERGY CORRELATION
# AVG -0.4412655 STD 0.1739638 MIN_R 0.8365040 MAX_R 14.4771496 AVGESTATIC
-0.4730655
...
```

These values have to be inserted manually into the options file for importing to the coarse-grained system (see below). Apart from that, the file contains the spatial correlation function.

You generate energies following this distribution and correlation by using the *eimport* calculator

**Import the energy distribution and correlation and reproduce it in stochastic network**

```
xtp_run -o options.xml -f state_ref.sql -e eimport
```

with the options:

**options_import.xml**

```
<options>
        <eimport>
                <probabilityfile_h>reference/eanalyze.sitecorr_h.out</probabilityfile_h>
                <sigma_h>0.1763163</sigma_h>
                <avgestatic_h>-0.5913265</avgestatic_h>
                <probabilityfile_e>reference/eanalyze.sitecorr_e.out</probabilityfile_e>
                <sigma_e>0.1739638</sigma_e>
                <avgestatic_e>-0.4730655</avgestatic_e>
                <cutoff>8.5</cutoff>
                <seed>1</seed>
        </eimport>
</options>
```

The *cutoff* keyword can be used to read in the correlation function only up to a certain distance, which can be useful if larger distances yield unphysical results.

### Transfer Integrals

The last ingredient reproduced by the stochastic approach are transfer integrals $J$. The idea is that $\log_{10}(J^2/\,\mathrm{eV}^2)$ is roughly Gaussian distributed, with mean and error of the distribution varying with distance (see figure 2.6 (d)). Use the calculator

**Analyze the distance-depend distribution of transfer integrals in the reference system**

```
xtp_run -o options.xml -f state_ref.sql -e ianalyze
```

with options

**options_analyze.xml**

```
<options>
        <ianalyze>
                <resolution_logJ2>0.05</resolution_logJ2>
                <resolution_space>0.05</resolution_space>
                <states>1,-1</states> <!-- +1 for hole transport, -1 for electron transport --
        </ianalyze>
</options>
```

That will generate the files **ianalyze.ispatial_e.out** and **ianalyze.ispatial_h.out**, which contain means and errors as a function of centre-of-mass distance.

Now use the *iimport* calculator to generate transfer integrals in the coarse grained state file, following the same statistics.

**Import distance dependent distribution of transfer integrals and reproduce in stochastic network**

```
xtp_run -o options.xml -f state_cg.sql -e iimport
```

**options_import.xml**

```
<options>
        <iimport>
                <TI_tag></TI_tag>
                <TI_file></TI_file>
                <idft_jobs_file></idft_jobs_file>
                <probabilityfile_h>reference/ianalyze.ispatial_h.out</probabilityfile_h>
                <probabilityfile_e>reference/ianalyze.ispatial_e.out</probabilityfile_e>
        </iimport>
</options>
```

### einternal

Run the *einternal* calculator, just as you do it for the reference system.

### Rates

If you followed the steps is this section, you have everything at hand to calculate charge transfer rates for the coarse grained system from the stochastic ingredients:

**Calculate rates in the stocchastic network**

```
xtp_run -o options.xml -f state_cg.sql -e rates
```

Options are the same as for the reference file. You can check the result by comparing rates from your reference to the coarse-grained system, see figure 2.6(e) for an example. The resulting charge transport network can be used for kinetic Monte Carlo simulations with VOTCA. If everything goes well, mobilities for both systems should agree, as shown in figure 2.6(f).

## 2.11 Macroscopic observables

sec:analysis

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 $O_\alpha$ in a state $\alpha$, its ensemble average at time $t$ is a sum over all states weighted by the probability $P_\alpha$ to be in a state $\alpha$ at time $t$

$$\langle O \rangle = \sum_\alpha O_\alpha P_\alpha. \tag{2.36}$$

equ:ensemble

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

$$\frac{d\langle O \rangle}{dt} = \sum_{\alpha,\beta} \left[ P_\beta \Omega_{\beta\alpha} - P_\alpha \Omega_{\alpha\beta} \right] O_\alpha = \sum_{\alpha,\beta} P_\beta \Omega_{\beta\alpha} \left[ O_\alpha - O_\beta \right]. \tag{2.37}$$

If averages are obtained from KMC trajectories, $P_\alpha = s_\alpha / s$, where $s_\alpha$ is the number of Markov chains ending in the state $\alpha$ 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 $\alpha$ is $\tau_\alpha$ then

$$\overline{O} = \frac{1}{\tau} \sum_\alpha O_\alpha \tau_\alpha \,, \tag{2.38}$$

equ:time

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.

### 2.11.1 Charge density

sec:occupation

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

$$\rho_i = e p_i / V_i \,, \tag{2.39}$$

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 sxtpthing kernel function, i.e. a distance-weighted average over multiple sites. Site occupations $p_i$ can be obtained from eq. (2.36) or eq. (2.38) by using the occupation of site $i$ in state $\alpha$ 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

$$p_j \omega_{ji} = p_i \omega_{ij}, \tag{2.40}$$

equ:detailed_balance

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. (2.31)).

### 2.11.2 Current

sec:vaverage

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

$$\vec{J} = e \langle \vec{v} \rangle = e \frac{d \langle \vec{r} \rangle}{dt} = e \sum_{i,j} p_j \omega_{ji} (\vec{r}_i - \vec{r}_j). \tag{2.41}$$ equ:current_def

Symmetrizing this expression we obtain

$$\vec{J} = \frac{1}{2} e \sum_{i,j} \left( p_j \omega_{ji} - p_i \omega_{ij} \right) \vec{r}_{ij}, \tag{2.42}$$ equ:current

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

$$\vec{J}_i = \frac{1}{2} e \sum_{j} \left( p_j \omega_{ji} - p_i \omega_{ij} \right) \vec{r}_{ij}. \tag{2.43}$$ equ:site_current

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 [44].

### 2.11.3 Mobility and diffusion constant

sec:mobility

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

$$2dD_{\gamma\delta} = \frac{d \langle \Delta r_{i,\gamma} \Delta r_{i,\delta} \rangle}{dt} = \sum_{\substack{i,j \\ i \neq j}} p_j \omega_{ji} \left( \Delta r_{i,\gamma} \Delta r_{i,\delta} - \Delta r_{j,\gamma} \Delta r_{j,\delta} \right) = \sum_{\substack{i,j \\ i \neq j}} p_j \omega_{ji} \left( r_{i,\gamma} r_{i,\delta} - r_{j,\gamma} r_{j,\delta} \right) . \tag{2.44}$$ equ:diffusion

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,

$$D_{\gamma\delta} = k_{\mathrm{B}} T \mu_{\gamma\delta} \,, \tag{2.45}$$

one can, in principle, obtain the zero-field mobility tensor $\mu_{\gamma\delta}$. Eq. (2.44), 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. (2.41)

$$\langle \vec{v} \rangle = \sum_{i,j} p_j \omega_{ji} (\vec{r}_i - \vec{r}_j) = \hat{\mu} \vec{F} \,, \tag{2.46}$$

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.

### 2.11.4 Spatial correlations of energetic disorder

sec:eanalyze

Long-range, e.g. electrostatic and polarization, interactions often result in spatially correlated disorder [45], which affects the onset of the mobility-field (Poole-Frenkel) dependence [40, 46, 47]. To quantify the degree of correlation, one can calculate the spatial correlation function of $E_i$ and $E_j$ at a distance $r_{ij}$

$$C(r_{ij}) = \frac{\langle (E_i - \langle E \rangle)\,(E_j - \langle E \rangle) \rangle}{\langle (E_i - \langle E \rangle)^2 \rangle}, \tag{2.47}$$ equ:cf

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 [48].

For systems with spatial correlations, variations in site energy differences, $\Delta E_{ij}$, 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 $\Delta E_{ij}$ 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**

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

# Chapter 3

# Input and output files

sec:io

## 3.1 Atomistic topology

sec:atomistic

If you are using GROMACS for generating atomistic configurations, it is possible to directly use the topology file provided by GROMACS (topology.tpr). In this case the GROMACS 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. Moreover, it s possible to partially overwrite the information provided in, for example, GROMACS topology file. We will illustrate how to create a custom topology file using DCV2T. The structure of DCV2T, together with atom type definitions, is shown in fig. 3.1. DCV2T has two thiophene (THI) and two dicyanovinyl (NIT) residues. The pdb file which contains residue types, residue numbering, atom names, atom types, and atom coordinates is shown in listing 3.1.

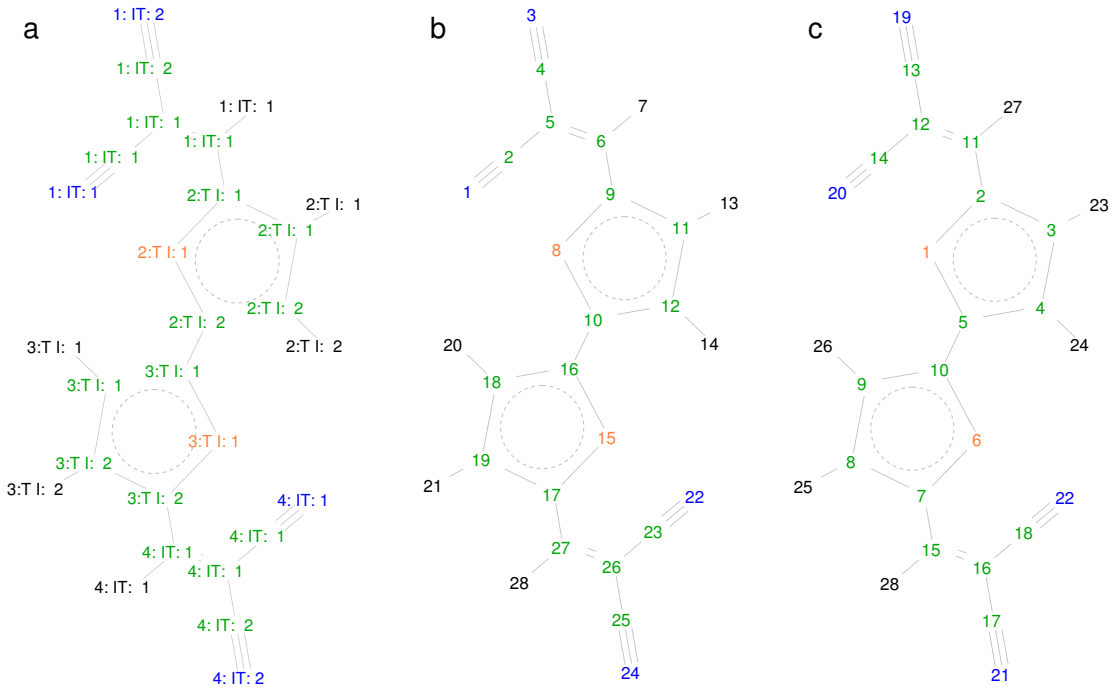



Figure 3.1: (a) DCV2T with atoms labelled according to residue_number:residue_name:atom_name. There are four residues and two residue types: thiophene (THI) and dicyanovinyl (NIT). The corresponding pdb file is shown in listing 3.1. Atom numbering is used to split conjugated segments on rigid fragments and to link atomistic ((b) from GROMACS topology) and quantum descriptions (c). fig:dcv2t

list:pdb

Listing 3.1: 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
```

tab:map

Table 3.1: Description of the XML mapping file (`map.xml`).

| | |
|---|---|
| topology | Definitions of molecules, segments, and fragments. |
| molecules | Container for all molecules. |
| molecule | Mapping of a single molecule. |
| name | Name of the molecule in the coarse-grained model. |
| ident | Name (identification) of the molecule in the all-atom representation. This must match the molecule name in the atomistic representation. |
| segments | Partitioning of the molecule on conjugated segments. |
| segment | Description of a conjugated segment. |
| name | Name of a conjugated segment in a molecule. |
| fragments | Container for all fragments in a segment. |
| fragment | Description of a rigid fragment. |
| name | Name of the rigid fragment in a conjugated segment |
| mdatoms | List of all atoms belonging to the rigid fragment in the format residue number:residue name:atom name. |
| qmatoms | List of atoms of the rigid fragment in its ground state geometry, atom number:atom type. |
| weights | Weights are used to determine the fragment center. The order should be the same as in the mdatoms and qmatoms definitions. If the mass of a nucleus in atomic mass units is used, the center of the rigid fragment will be its center of mass. |
| localframe | Three atoms which define a local frame for each rigid fragment. |

## 3.2 Mapping file

sec:xmlmap

The mapping file (referred here as `map.xml`) is used by the program `xtp_map` to convert an atomistic trajectory to a trajectory with conjugated segments and rigid fragments. This trajectory is stored in a state file and contains positions, names, types of atoms belonging to rigid fragments. The description of the mapping options is given in table 3.1. An example of `map.xml` for a DCV2T molecule is shown in listing 3.2.
The file `map.xml` contains the whole electrostatic information about the molecules as well as the structural information. The toolpdb2map creates a `map.xml` from a pdb file and is a good starting point for further refinement.

list:map

Listing 3.2: 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 3.1 of the pdb file).

```
<topology> <!-- this file is used to conver an atomistic trajectory to conjugated segments -->
<molecules>
<molecule>
   <name>DCV2T-MOL</name> <!-- name of the conjugated molecule -->
   <mdname>Protein</mdname> <!-- corresponding name of this molecule in the MD trajectory, should be
        the same as the name given at the end of 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 -->
        <basisset>INDO</basisset>
        <orbitals>QC_FILES/DCV2T.orb</orbitals>
        <torbital_h>50</torbital_h><!-- Number of the HOMO Orbital (e.g.  alpha electrons, can be
             found in the log-file belonging to DCV2T.orb) -->

            <!-- EMULTIPOLE INPUT -->
        <multipoles_n>MP_FILES/DCV2T.mps</multipoles_n><!-- Multipole file for neutral state -->
        <multipoles_h>MP_FILES/DCV2T_h.mps</multipoles_h><!-- Multipole file for hole state -->
        <map2md>0</map2md><!-- specifies if planar QM coordinates (map2md=0) or MD coordinates (
             map2md=1) of atoms are used for distribution of partial charges. For MD coordinates the
             order and numbering in <mdatoms> and  <mpoles> must be identical it has no impact on the
              qm e.g. DFT or GWBSE calculations-->

                <!-- 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 Mapping -->
        <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 cartesian local frame, two atoms: fragment is assumed to be
             rotationally invariant around the axis, one atom: fragment is assumed isotropic -->
         <localframe> 20 19 14 </localframe>
        <!-- Optional parameters (if not set <localframe> is used): used when atom labels in the .mps
              and .xyz file differ or more sites in the .mps file are used, so refers to <mpoles> -->
         <localframe_mps> 20 19 14 </localframe_mps>
         <!-- Optional parameters (if not set <localframe> is used):  weights to determine the
             fragment center (here CoM is used), used when atom labels in the .mps and .xyz file
              differ or additional sites in the .mps file are used -->
        <weights_mps>     14       12        14        12        12        12         1  </
            weights_mps>
        <!-- Optional flag: says  if a site is virtual or not, (virtual=1, 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>
```

## 3.3 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 3.3. 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.

**Be careful!**

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

list:zindo_orbitals

Listing 3.3: `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
```

## 3.4 Monomer calculations for DFT transfer integrals

list:edft_gaussian_xml

Listing 3.4: 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>
```

list:edft_turbomole_xml

Listing 3.5: 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>
```

list:edft_nwchem_xml

Listing 3.6: 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>
```

## 3.5 Pair calculations for DFT transfer integrals

list:idft_gaussian_xml

Listing 3.7: 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=
  <memory>1Gb</memory>
  <threads>1</threads>

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

list:idft_turbomole_xml

Listing 3.8: 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>
```

list:idft_nwchem_xml

Listing 3.9: 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>
```

## 3.6 DFT transfer integrals

list:TI_xml

Listing 3.10: 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>
```

## 3.7 State file

sec:statefile

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_coulomb\""
```

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\""
```

# Chapter 4

# Reference

sec:reference

## 4.1 Programs

sec:programs

Programs execute specific tasks (calculators).

### 4.1.1 xtp_testsuite

prog:xtp_testsuite

Performs tests en suite + optional arguments:

`-h, --help` show this help message and exit
`-e [ [ ...]], --execute [ [ ...]]` Tests to perform, accepts regex (def=".*")
`-l, --listonly` List all tests available, then quit.
`-x , --xml` Test-suite file (def="$VOTCASHARE/xtp/xml/testsuite.xml")
`-s , --source` Test source input directory (def="source")
`-td , --testdirectory` Test run directory (def="suite")
`-t , --target` Directory where to store targets (def="targets")
`-r , --reference` Folder with reference data to compare to (def="reference")
`-g, --generate` Generate reference from targets (def=False)
`-cmp, --compareonly` Only compare existing targets to reference (def=False)
`-v, --verbose` The wordy version (def=False)
`-sh, --showoutput` Display VOTCA::XTP exec. output (def=False)
`-c, --clean` To clean or not to clean test dir. (def=False)
`-m , --mailto` Mail the result. (def=False)

### 4.1.2 xtp_update

prog:xtp_update

Updates the state file + optional arguments:

`-h, --help` show this help message and exit
`-f SQLFILE, --file SQLFILE` State file to update.

### 4.1.3 xtp_update_exciton

prog:xtp_update_exciton

Updates the state file for singlets and triplets + optional arguments:

`-h, --help` show this help message and exit
`-f SQLFILE, --file SQLFILE` State file to update.

### 4.1.4 xtp_basisset

prog:xtp_basisset

xtp_update, version 1.4_rc1 Creates votca xml basissetfiles from NWCHEM basissetfiles optional arguments:

`-h, --help` show this help message and exit
`-f NWCHEM, --inputnw NWCHEM` NWchem file containing the basisset.
`-o OUTPUTFILE, --outputvotca OUTPUTFILE` Path of votca outputfile

### 4.1.5 xtp_map

prog:xtp_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

### 4.1.6 xtp_run

prog:xtp_run

Runs excitation/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

### 4.1.7 xtp_tools

prog:xtp_tools

Runs excitation/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 Tools:
`-e [ --execute ] arg` List of tools separated by ',' or ' '
`-l [ --list ]` Lists all available tools
`-d [ --description ] arg` Short description of a tool

### 4.1.8 xtp_parallel

prog:xtp_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

### 4.1.9 xtp_dump

prog:xtp_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 Extractors:
`-e [ --extract ] arg` List of extractors separated by ',' or ' '
`-l [ --list ]` Lists all available extractors
`-d [ --description ] arg` Short description of an extractor

### 4.1.10 xtp_overlap

prog:xtp_overlap

moo_overlap
`-h [ --help ]` display this help and exit
`-v [ --verbose ]` be loud and noisy MOO Options:
`--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

### 4.1.11 xtp_kmc_run

prog:xtp_kmc_run

kmc_run, version 1.4_rc1 (compiled Sep 26 2016, 22:47:52) 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

## 4.2 Calculators

sec:calculators

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

```
xtp_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 4.1. A complete list of calculators is given in the calculators reference section.

list:calc

Listing 4.1: 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

```
xtp_run --list
```

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

```
xtp_run --desc calc1,calc2,...
```

### 4.2.1 coupling

calc:coupling

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

| option | default | unit | description |
|---|---|---|---|
| dftpackage | | | First-principles package |
| output | coupling.out.xm | | 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`.

### 4.2.2 excitoncoupling

calc:excitoncoupling

Exciton couplings from serialized orbital files

| option | default | unit | description |
|---|---|---|---|
| classical | | | |
| output | excitoncoupling | | Output file |
| bsecoupling_options | | | |
| orbitalsA | A.orb | | Serialized orbitals file |
| orbitalsB | B.orb | | Serialized orbitals file |
| orbitalsAB | AB.orb | | Serialized orbitals file |

Return to the description of `excitoncoupling`.

### 4.2.3 gencube

calc:gencube

Tool to generate cube files from .orb file

| option | default | unit | description |
|---|---|---|---|
| output | state.cube | | Output file |
| input | system.orb | | Input file |
| padding | 6.5 | | How far the grid should start from the molecule |
| xsteps | 25 | | Gridpoints in x-direction |
| ysteps | 25 | | Gridpoints in y-direction |
| zsteps | 25 | | Gridpoints in z-direction |
| state | 1 | | State to generate cube file for |
| spin | | | Singlet or Triplet |
| type | ground | | qp:quasiparticle,ground:groundstate,transition:transitionstate,excited/exc gs:excitedstate density/density excited-ground state |
| mode | new | | new: generate new cube file, substract: substract to cube files specified below |
| infile1 | | | Cubefile to substract infile2 from |
| infile2 | | | Cubefile to substract from infile1 |

Return to the description of `gencube`.

### 4.2.4 log2mps

calc: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`.

### 4.2.5 molpol

calc: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`.

### 4.2.6 orb2isogwa

calc:orb2isogwa

Analysis tool for QM results stores in serialized file

| option | default | unit | description |
|---|---|---|---|
| output | qmanalyze.out | | Output file |
| property | | | |
| input | molecule.orb | | Serialized file |

Return to the description of `orb2isogwa`.

### 4.2.7 partialcharges

calc:partialcharges

Tool to derive partial charges from QM results stores in serialized file

| option | default | unit | description |
|---|---|---|---|
| output | Moleculecharge | | Output file either .mps or .pdb |
| input | molecule.orb | | Serialized file |
| esp_options | | | options for the method |

Return to the description of `partialcharges`.

### 4.2.8 pdb2map

calc: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`.

### 4.2.9 pdb2top

calc: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`.

### 4.2.10 ptopreader

calc: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`.

### 4.2.11 qmanalyze

calc:qmanalyze

Analysis tool for QM results stores in serialized file

| option | default | unit | description |
|---|---|---|---|
| output | qmanalyze.out | | Output file |
| BSE | | | additonal info about BSE results |
| input | molecule.orb | | Serialized file |

Return to the description of `qmanalyze`.

### 4.2.12 eanalyze

calc: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`.

### 4.2.13 eimport

calc: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`.

### 4.2.14 einternal

calc: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`.

### 4.2.15 emultipole

calc: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`.

### 4.2.16 eoutersphere

calc: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`.

### 4.2.17 ianalyze

calc:ianalyze

Evaluates a histogram of a logarithm of squared couplings

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

Return to the description of `ianalyze`.

### 4.2.18 iimport

calc:iimport

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

| option | default | unit | description |
|---|---|---|---|
| idft_jobs_file | | | idft jobs file |
| probabilityfile_h | ianalyze.ispatial h.out | | For coarse grained simulations provide here the distance dependent means and sigmas of hole transfer integrals. This file can be created using the ianalyze calculator. |
| probabilityfile_e | ianalyze.ispatial e.out | | For coarse grained simulations provide here the distance dependent means and sigmas of electron transfer integrals. This file can be created using the ianalyze calculator. |

Return to the description of `iimport`.

### 4.2.19 izindo

calc: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`.

### 4.2.20 jobwriter

calc:jobwriter

Writes list of jobs for a parallel execusion

| option | default | unit | description |
|---|---|---|---|
| keys | | | job type |
| 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`.

### 4.2.21 pairdump

calc: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`.

### 4.2.22 panalyze

calc:panalyze

Probability of neighbours being in the pair list as a function of their centre of mass distance

| option | default | unit | description |
|---|---|---|---|
| resolution_space | | nm | Spatial resolution for the probability function. |

Return to the description of `panalyze`.

### 4.2.23 profile

calc: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`.

### 4.2.24 rates

calc: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`.

### 4.2.25 sandbox

calc:sandbox

Sandbox to test xtp classes

| option | default | unit | description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| ID | | | Not in use |

Return to the description of `sandbox`.

### 4.2.26 stateserver

calc: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`.

### 4.2.27 tdump

calc: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`.

### 4.2.28 vaverage

calc: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`.

### 4.2.29 zmultipole

calc: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`.

### 4.2.30 edft

calc:edft

A wrapper for first principles based single site calculations

| option | default | unit | description |
|---|---|---|---|
| tasks | input,run,parse | | What to run |
| store | orbitals | | What to store |

Return to the description of `edft`.

### 4.2.31 idft

calc:idft

Projection method for electronic couplings. Requires edft otput

| option | default | unit | description |
|---|---|---|---|
| tasks | input,run,parse, | | What to do |
| store | orbitals,overlap | | 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`.

### 4.2.32 qmmm

calc:qmmm

QM/MM with the Thole MM model

| option | default | unit | description |
|---|---|---|---|
| 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 |
| dftpackage | | | DFT 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`.

### 4.2.33 xqmultipole

calc:xqmultipole

Electrostatic interaction and induction energy of charged molecular clusters

| option | default | unit | description |
|---|---|---|---|
| mapping | | | Polar-site mapping definition |
| 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`.

### 4.2.34 energy2xml

calc:energy2xml

Write out energies from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `energy2xml`.

### 4.2.35 integrals2xml

calc:integrals2xml

Write out transfer integrals from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `integrals2xml`.

### 4.2.36 occupations2xml

calc:occupations2xml

Write out site occupation probabilities from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `occupations2xml`.

### 4.2.37 pairs2xml

calc:pairs2xml

Write out neighbourlist from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `pairs2xml`.

### 4.2.38 rates2xml

calc:rates2xml

Write out charge transfer rates from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `rates2xml`.

### 4.2.39 segments2xml

calc:segments2xml

Write out segment data from SQL file

| option | default | unit | description |
|---|---|---|---|

Return to the description of `segments2xml`.

### 4.2.40 trajectory2pdb

calc:trajectory2pdb

Generate PDB files for the mapped MD/QM topology

| option | default | unit | description |
|---|---|---|---|

Return to the description of `trajectory2pdb`.

## 4.3 Common options

ref:options

| name | Description of the option |
|---|---|

# Bibliography


ruhle_microscopic_2011 [1] V. RÃijhle *et al.*, J. Chem. Theory Comput. **7**, 3335 (2011), bibtex: ruhle_microscopic_2011. ii, 3, 19

ruhle_versatile_2009 [2] V. RÃijhle *et al.*, J. Chem. Theory Comput. **5**, 3211 (2009), bibtex: ruhle_versatile_2009. ii, 3, 6

gamma_design_1995 [3] E. Gamma, R. Helm, and R. E. Johnson, *Design Patterns. Elements of Reusable Object-Oriented Software.*, 1st ed., reprint. ed. (Addison-Wesley Longman, Amsterdam, ADDRESS, 1995), bibtex: gamma_design_1995. 3

ruhle_multiscale_2010 [4] V. RÃijhle, J. Kirkpatrick, and D. Andrienko, J. Chem. Phys. **132**, 134103 (2010), bibtex: ruhle_multiscale_2010. 7

vukmirovic_charge_2008 [5] N. VukmiroviÄĞ and L.-W. Wang, J. Chem. Phys. **128**, 121102 (2008), bibtex: vukmirovic_charge_2008. 7

vukmirovic_charge_2009 [6] N. VukmiroviÄĞ and L.-W. Wang, Nano Letters **9**, 3996 (2009), bibtex: vukmirovic_charge_2009.

mcmahon_ad_2009 [7] D. P. McMahon and A. Troisi, Chem. Phys. Lett. **480**, 210 (2009), bibtex: mcmahon_ad_2009. 7

bredas_charge-transfer_2004 [8] J.-L. BrÃĺdas, D. Beljonne, V. Coropceanu, and J. Cornil, Chem. Rev. **104**, 4971 (2004), bibtex: bredas_charge-transfer_2004. 9, 19

may_relationship_2011 [9] F. May *et al.*, J. Mater. Chem. **21**, 9538 (2011), bibtex: may_relationship_2011. 9, 20

breneman_determining_1990 [10] C. M. Breneman and K. B. Wiberg, J. Comput. Chem. **11**, 361 (1990), bibtex: breneman_determining_1990. 11

stone_distributed_1985 [11] A. Stone and M. Alderton, Molecular Physics **56**, 1047 (1985), bibtex: stone_distributed_1985. 11, 12

chirlian_atomic_1987 [12] L. E. Chirlian and M. M. Francl, J. Comput. Chem. **8**, 894 (1987), bibtex: chirlian_atomic_1987. 11

singh_approach_1984 [13] U. C. Singh and P. A. Kollman, J. Comput. Chem. **5**, 129 (1984), bibtex: singh_approach_1984. 11

stone_distributed_2005 [14] A. J. Stone, J. Chem. Theory Comput. **1**, 1128 (2005), bibtex: stone_distributed_2005. 11, 12

stone_theory_1997 [15] A. J. Stone, *The Theory of intermolecular forces* (Clarendon Press, Oxford, 1997), bibtex: stone_theory_1997. 12

thole_molecular_1981 [16] B. Thole, Chem. Phys. **59**, 341 (1981), bibtex: thole_molecular_1981. 13

van_duijnen_molecular_1998 [17] P. T. van Duijnen and M. Swart, J. Phys. Chem. A **102**, 2399 (1998), bibtex: van_duijnen_molecular_1998-1. 13

applequist_atom_1972 [18] J. Applequist, J. R. Carl, and K.-K. Fung, J. Am. Chem. Soc. **94**, 2952 (1972), bibtex: applequist_atom_1972. 13

ren_polarizable_2003 [19] P. Ren and J. W. Ponder, J. Phys. Chem. B **107**, 5933 (2003), bibtex: ren_polarizable_2003. 13

baessler_charge_1993 [20] H. Baessler, Phys. Status Solidi B **175**, 15 (1993), bibtex: baessler_charge_1993. 14, 19

troisi_charge-transport_2006 [21] A. Troisi and G. Orlandi, Phys. Rev. Lett. **96**, (2006), bibtex: troisi_charge-transport_2006.

troisi_charge_2009 [22] A. Troisi, D. L. Cheung, and D. Andrienko, Phys. Rev. Lett. **102**, 116602 (2009), bibtex: troisi_charge_2009.

mcmahon_organic_2010 [23] D. P. McMahon and A. Troisi, ChemPhysChem **11**, 2067 (2010), bibtex: mcmahon_organic_2010.

vehoff_charge_2010 [24] T. Vehoff *et al.*, J. Phys. Chem. C **114**, 10592 (2010), bibtex: vehoff_charge_2010. 14

baumeier_density-functional_2010 [25] B. Baumeier, J. Kirkpatrick, and D. Andrienko, Phys. Chem. Chem. Phys. **12**, 11103 (2010), bibtex: baumeier_density-functional_2010. 14, 17, 19

kirkpatrick_approximate_2008 [26] J. Kirkpatrick, Int. J. Quantum Chem. **108**, 51 (2008), bibtex: kirkpatrick_approximate_2008. 18

walker_electrical_2002 [27] A. B. Walker, A. Kambili, and S. J. Martin, J. Phys-Condens. Mat. **14**, 9825 (2002), bibtex: walker_electrical_2002. 19

borsenberger_charge_1991 [28] P. M. Borsenberger, L. Pautmeier, and H. BÃd'ssler, J. Chem. Phys. **94**, 5447 (1991), bibtex: borsenberger_charge_1991.

pasveer_unified_2005 [29] W. F. Pasveer *et al.*, Phys. Rev. Lett. **94**, 206601 (2005), bibtex: pasveer_unified_2005. 19

bredas_molecular_2009 [30] J.-L. BrÃľdas, J. E. Norton, J. Cornil, and V. Coropceanu, Accounts Chem. Res. **42**, 1691 (2009), bibtex: bredas_molecular_2009. 19

coropceanu_charge_2007 [31] V. Coropceanu *et al.*, Chem. Rev. **107**, 926 (2007), bibtex: coropceanu_charge_2007.

nelson_modeling_2009 [32] J. Nelson, J. J. Kwiatkowski, J. Kirkpatrick, and J. M. Frost, Accounts Chem. Res. **42**, 1768 (2009), bibtex: nelson_modeling_2009. 19

marcus_electron_1993 [33] R. A. Marcus, Rev. Mod. Phys. **65**, 599 (1993), bibtex: marcus_electron_1993. 19

hutchison_hopping_2005 [34] G. R. Hutchison, M. A. Ratner, and T. J. Marks, J. Am. Chem. Soc. **127**, 2339 (2005), bibtex: hutchison_hopping_2005. 19

chang_new_2005 [35] J.-L. Chang, J. Mol. Spectrosc. **232**, 102 (2005), bibtex: chang_new_2005. 20

hoffman_reorganization_1996 [36] B. M. Hoffman and M. A. Ratner, Inorg. Chim. Acta. **243**, 233 (1996), bibtex: hoffman_reorganization_1996. 20

bortz_new_1975 [37] A. B. Bortz, M. H. Kalos, and J. L. Lebowitz, J. Comput. Phys. **17**, 10 (1975), bibtex: bortz_new_1975. 21

scher_anomalous_1975 [38] H. Scher and E. Montroll, Phys. Rev. B **12**, 2455 (1975), bibtex: scher_anomalous_1975. 21

borsenberger_role_1993 [39] P. M. Borsenberger, E. H. Magin, M. D. VanAuweraer, and F. C. D. Schryver, Phys. Status Solidi A **140**, 9 (1993), bibtex: borsenberger_role_1993. 21

derrida_velocity_1983 [40] B. Derrida, J. Stat. Phys. **31**, 433 (1983), bibtex: derrida_velocity_1983. 21, 31

cordes_one-dimensional_2001 [41] H. Cordes *et al.*, Phys. Rev. B **63**, 094201 (2001), bibtex: cordes_one-dimensional_2001.

seki_electric_2001 [42] K. Seki and M. Tachiya, Phys. Rev. B **65**, 014305 (2001), bibtex: seki_electric_2001. 21

lukyanov_extracting_2010 [43] A. Lukyanov and D. Andrienko, Phys. Rev. B **82**, 193202 (2010), bibtex: lukyanov_extracting_2010. 21

van_der_holst_modeling_2009 [44] J. J. M. van der Holst *et al.*, Phys. Rev. B **79**, 085203 (2009), bibtex: van_der_holst_modeling_2009. 30

dunlap_charge-dipole_1996 [45] D. Dunlap, P. Parris, and V. Kenkre, Phys. Rev. Lett. **77**, 542 (1996), bibtex: dunlap_charge-dipole_1996. 31

novikov_essential_1998 [46] S. V. Novikov *et al.*, Phys. Rev. Lett. **81**, 4472 (1998), bibtex: novikov_essential_1998. 31

nagata_atomistic_2008 [47] Y. Nagata and C. Lennartz, J. Chem. Phys. **129**, 034709 (2008), bibtex: nagata_atomistic_2008. 31

novikov_cluster_1995 [48] S. V. Novikov and A. V. Vannikov, J. Phys. Chem. **99**, 14573 (1995), bibtex: novikov_cluster_1995. 31