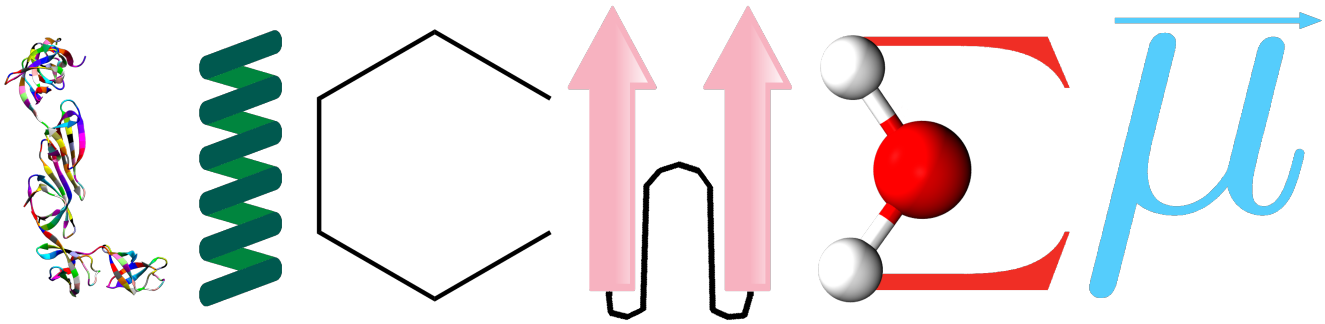
LICHEM: Layered Interacting CHEmical Models
Symbiotic Computational Chemistry
v1.1



A QM/MM Interface for Polarizable Force Fields

Eric G. Kratz, Hatice Gökcan, Alice Walker, Erik A. Vázquez–Montelongo,
G. Andrés Cisneros,
2018

Contents

1 Introduction **3**
- 1.1 Design principles . . . 3
- 1.2 Notes for developers . . . 3
  - 1.2.1 Language and style . . . 3
  - 1.2.2 Updating LICHEM . . . 4
  - 1.2.3 Parallelization . . . 4
- 1.3 Installation . . . 4
- 1.4 LICHEM test suite . . . 5
- 1.5 Capabilities . . . 6
  - 1.5.1 QM and MM wrappers . . . 6
  - 1.5.2 Types of calculations . . . 6
- 1.6 Acknowledgements . . . 6

2 LICHEM Input **9**
- 2.1 Command line arguments . . . 9
  - 2.1.1 Load Balancing with `lichem.MPI` . . . 10
- 2.2 XYZ input files . . . 10
- 2.3 Connectivity input files . . . 11
- 2.4 Region input files . . . 12
  - 2.4.1 Keywords . . . 12
- 2.5 Calculation types and synonyms . . . 18
  - 2.5.1 Single-point energies . . . 18
  - 2.5.2 Optimizations . . . 18
  - 2.5.3 Reaction Paths and Restrained MM Optimization . . . 18
  - 2.5.4 Ensemble sampling . . . 19
- 2.6 File conversion . . . 20
- 2.7 LICHEM output . . . 21
- 2.8 Additional input for wrappers . . . 21
  - 2.8.1 TINKER . . . 21
  - 2.8.2 LAMMPS . . . 22
  - 2.8.3 Gaussian . . . 22
  - 2.8.4 NWChem . . . 22
  - 2.8.5 Restart files . . . 22
  - 2.8.6 Examples . . . 23

3 Theoretical Background **24**
3.1 Preface . . . 24
3.2 Classical methods . . . 24
3.2.1 Newton's laws of motion . . . 24
3.2.2 Force fields . . . 25
3.2.3 Multipolar/polarizable force fields . . . 25
3.2.4 Polarizable density force fields . . . 26
3.3 Geometry and reaction path optimizations . . . 26
3.3.1 Steepest descent . . . 26
3.3.2 Davidon-Fletcher-Powell . . . 27
3.3.3 Nudged elastic band . . . 27
3.3.4 Quadradic string method . . . 29
3.4 Quantum methods . . . 29
3.4.1 Schrödinger equation . . . 29
3.4.2 Perturbation theory . . . 30
3.4.3 Variational theorem . . . 31
3.4.4 Basis sets . . . 31
3.4.5 Pseudopotentials . . . 32
3.4.6 Orbital-free density functional theory . . . 33
3.4.7 Self-consistent field methods . . . 34
3.4.8 Kohn-Sham DFT . . . 34
3.4.9 Dispersion corrections . . . 35
3.5 QM/MM . . . 35
3.5.1 QM-MM interactions . . . 36
3.5.2 Pseudobond method . . . 36
3.6 Monte Carlo simulations . . . 37
3.6.1 Stochastic sampling . . . 37
3.6.2 Canonical ensemble . . . 38
3.6.3 Isobaric-isothermal ensemble . . . 38
3.7 Path-integral Monte Carlo . . . 39
3.7.1 Path-integral formalism . . . 39
3.7.2 PIMC simulations . . . 39
3.7.3 Ergodicity . . . 39

Chapter 1

Introduction

1.1 Design principles

LICHEM is in development with four guiding principles:

1) **LICHEM should primarily be an interface.** In order to allow the code to communicate with a variety of (continuously updated) software packages, LICHEM for general applications should not require modifications to other software packages or tools.
2) **LICHEM should be a single executable.** Since LICHEM is an interface, there is no need for a large collection of binaries. Furthermore, LICHEM can be statically linked to ease distribution of the software.
3) **LICHEM should be easy to read and modify.** Most scientists are not experts in computer programming. In order to allow for rapid collaborative development of LICHEM, the source code needs to be easy to read. To that end, the code should be simple, well commented, and avoid linking to libraries.
4) **LICHEM should be open-source.** LICHEM is licensed under the GPLv3. Science can benefit everyone, and hence, should be shared as widely as possible.

1.2 Notes for developers

1.2.1 Language and style

The LICHEM repository contains code written in C++, LaTeX, python, and markdown. All lines of code should be less than 80 characters wide. This has nothing to do with compiling or tracking changes, but helps to make the code readable on smaller portable devices such as tablets and netbooks. Also for readability purposes, tabulars should not be used to indent the code. Tabulars can appear to be different sizes in some text editors.

When updating the LaTeX documentation, there should be a newline after each sentence. This makes the documentation files very long, but it makes tracking/merging changes safer and easier. Paragraphs are ended with a LaTeX newline followed by a blank line.

1.2.2 Updating LICHEM

Most of the development of LICHEM should be done though git (or other version control packages) and merged into the main repository. Since LICHEM does not require a large number of libraries or proprietary software, it should be easy to share new wrappers and approximations with the community. Additionally, new wrappers and approximations should be placed into separate files.

When functionality is added to LICHEM, the goal should be to make it simple and robust. In general we prefer to avoid adding lots of input keywords and extra input files, as much as possible. Most of the wrappers should be developed with the assumption that the QM/MM calculations will be large and quite difficult to simulate.
LICHEM should be tested with extremely difficult systems. During the earliest stages of the development, test systems with highly distorted structures were employed to make sure that crashes were rare.

A simple way to avoid modifying the QM and MM packages is to search for iterative or perturbative approximations for the QM/MM equations. This is extremely useful for multipolar/polarizable force fields.

1.2.3 Parallelization

LICHEM is primarily parallelized with OpenMP, with the exception of the Quadratic String Method (QSM), which can be used in a hybrid–parallel environment (MPI/OpenMP). The packages called by the wrappers can use any form of parallelization. Special care must be taken to avoid creating too many threads when calling multiple hybrid–parallel functions with the wrappers.

In NEB and OpenMP–QSM (OpenMP) simulations the number of CPUs given as a command line argument is used to divide the OpenMP threads between the replicas. In MPI QSM simulations (hybrid–parallel), the number of CPUs given as a command line argument define the number of available nodes for MPI.

1.3 Installation

Currently, the binary and user's manual are not included in the repository. However, the Makefile can be used to generate both files. Since LICHEM is designed to be simple, only a small number of packages are required to compile the code. An approximate list of packages is given below.

LICHEM binary: OpenMP (MPI if hybrid–parallel desired)
LICHEM manual: LaTeX, BibTex, TeXLive

A copy of the Eigen3 library is included with the LICHEM source code. However, other versions of Eigen3 can be used to build LICHEM by modifying the Makefile.

In order to install the OpenMP version of LICHEM, the user should first configure the program by:
`user:$ ./configure`

The configuration process will create a Makefile to install the OpenMP version of LICHEM under the installation directory (i.e. LICHEMvXX/bin/). After configuration, LICHEM can be installed by;

```
user:$ make install
```

If the hybrid–parallel version of LICHEM (lichem.MPI) is required, the configuration should be performed with;

```
user:$ ./configure --parallel
```

Both OpenMP and hybrid–parallel versions of LICHEM (`lichem` and `lichem.MPI` respectively), will be installed in the same directory if a specific installation directory is not specified during configuration;

```
user:$ ./configure --prefix=install_directory.
```

On Ubuntu boxes, the Makefile should function without modifications. However, it may be necessary to install additional LaTeX packages. On OSX machines, the SEDI, TEX, BIB, and CXXFLAGS variables will need to be modified.

The Makefile can produce both the documentation and the binary.

For development, debugging, or testing an alternate binary can be compiled.

```
user:$ make Dev
```

Additional make rules can be found in the Makefile.

QM and MM packages are installed separately from LICHEM. Users only need to install the packages that they wish to use in the QM/MM calculations.

1.4 LICHEM test suite

Test calculations can be performed with the runtests script in the tests directory. The test suite is semi-automated and will search the user's path for QM and MM packages.

Tests can be performed for pairs of QM and MM wrappers,

```
user:$ ./runtests [Ncpus] [QMPackage] [MMPackage]
```

or for all wrappers at once.

```
user:$ ./runtests [Ncpus] All
```

A dry run can be performed to check packages without performing the calculations.

```
user:$ ./runtests [Ncpus] [QMPackage] [MMPackage] Dry
```

Dry runs are useful for checking which QM and MM packages were found in the path.

Descriptsions of the tests can be found in Table 1.1. If tests are consistently failing, please post details on the GitHub issues section.

1.5 Capabilities

1.5.1 QM and MM wrappers

LICHEM can perform QM, MM, or QM/MM calculations via an interface to packages in the user's path. Temporary input files are created for each package and the results are collected from the temporary output files. This procedure is less efficient than linking to the packages directly, however, reading and writing input/output files is often negligible compared to the computational cost of the QM calculation. Currently, calculations can be performed using Gaussian [1], PSI4 [2], NWChem [3], TINKER [4], and LAMMPS [5]. Additional wrappers can be added to LICHEM with relatively little effort.

1.5.2 Types of calculations

Single-point energies, geometry optimizations, steepest descent minimizations, reaction pathways, ensemble sampling, classical Monte Carlo, and path-integral Monte Carlo calculations can be performed using the QM and MM wrappers. Due to limitations of the software packages, not all calculations can currently be performed with all combinations of wrappers. For convenience, the capabilities are summarized in Table 1.2 and 1.3.

Some additional restrictions are also present when there are bonds between the QM and MM regions. Currently, PSI4 cannot be used as a QM wrapper for calculations where the QM and MM regions are bonded.

1.6 Acknowledgements

The development of LICHEM was supported by funding from the NIH (Grant No. R01GM108583), Wayne State University, and the University of North Texas. LICHEM is maintained by the Cisneros research group at the University of North Texas.

Test	Description	QM	MM
HF energy	HF/6-31++G(d,p) energy of the water dimer calculated using only the QM wrapper	PSI4,Gaussian	N/A
PBE0 energy	PBE0/6-31++G(d,p) energy of the water dimer calculated using only the QM wrapper.	PSI4,Gaussian,NWChem	N/A
CCSD energy	CCSD/6-31++G(d,p) energy of the water dimer calculated using only the QM wrapper.	PSI4	N/A
PM6 energy	PM6 energy of the water dimer calculated using only the QM wrapper.	Gaussian	N/A
Frequencies	Harmonic frequencies of $[F–(CH_3)–F]^-$ calculated using only the QM wrapper.	PSI4,Gaussian,NWChem	N/A
NEB TS energy	Nudged elastic band optimization of $[F–(CH_3)–F]^-$ using only the QM wrapper.	PSI4,Gaussian,NWChem	N/A
TIP3P energy	MM energy of the water dimer with the TIP3P model.	N/A	TINKER
AMOEBA/GK energy	MM energy of the water dimer in the generalized Kirkwood implicit solvent.	N/A	TINKER
PBE0/TIP3P energy	QM/MM energy of a water dimer calculated with PBE0 and TIP3P.	PSI4,Gaussian,NWChem	TINKER
PBE0/AMOEBA energy	Polarizable QM/MM energy of a water dimer calculated with PBE0 and AMOEBA.	PSI4,Gaussian,NWChem	TINKER
DFP/Pseudobonds	QM/MM Davidon-Fletcher-Powell optimization of 2-Butyne with the two methyl groups replaced by pseudobond/boundary atoms.	Gaussian,NWChem	TINKER

Table 1.1: Approximate descriptions of the tests in the LICHEM test suite. Only some of the tests can be performed by the pairs of wrappers. More detailed descriptions can be found in the tests/README.md file.

	Gaussian	PSI4	NWChem
QM energy	Yes	Yes	Yes
QM/MM energy	Yes	Yes	Yes
QM opt.	Yes	Yes	No
QM/MM opt.	Yes	No	No
QM SD/DFP	Yes	Yes	Yes
QM/MM SD/DFP	Yes	Yes	Yes
QM MC	Yes	Yes	Yes
QM/MM MC	Yes	Yes	Yes
QM PIMC	Yes	Yes	Yes
QM/MM PIMC	Yes	Yes	Yes
QM RP	Yes	Yes	Yes
QM/MM RP	Yes	Yes	Yes

Table 1.2: QM wrapper capabilities for single-point energy, geometry optimization, steepest descent or Davidon-Fletcher-Powell (SD/DFP), Monte Carlo (MC), path-integral Monte Carlo (PIMC), and reaction path (RP) calculations.

	TINKER	LAMMPS
MM energy	Yes	No
QM/MM energy	Yes	No
MM opt.	Yes	No
QM/MM opt.	Yes	No
MM SD/DFP	No	No
QM/MM SD/DFP	Yes	No
MM MC	Yes	No
QM/MM MC	Yes	No
MM PIMC	Yes	No
QM/MM PIMC	Yes	No
MM RP	No	No
QM/MM RP	Yes	No

Table 1.3: MM wrapper capabilities for single-point energy, geometry optimization, steepest descent or Davidon-Fletcher-Powell (SD/DFP), Monte Carlo (MC), path-integral Monte Carlo (PIMC), and reaction path (RP) calculations.

Chapter 2

LICHEM Input

2.1 Command line arguments

LICHEM can only be invoked from a command line interface.

```
user:$ lichem -n Ncpus -x xyzfile.xyz -c confile.inp -r regfile.inp -o output.xyz
```

-n: Number of CPUs used in the calculations. Note that during PIMC and reaction path calculations each replica uses this many CPUs. A PIMC simulation with 8 beads and Ncpus=2 may require 16 CPUs.

-x: File name for the input structure in XYZ format. The XYZ input should be in the standard format and have a blank comment line.

-c: File name for connectivity and force field information.

-r: File name for definitions of QM/MM regions, QM wrapper, MM wrapper, and general simulation options.

-o: File name for trajectories and optimized structures.

-l: File name for output of LICHEM.

Currently, LICHEM can be executed in hybrid–parallel only for QSM. For hybrid–parallel execution of LICHEM;

```
mpirun -np Nprocs --hostfile host_list lichem.MPI
       -n NCpus
       -x react.xyz
       -c connect.inp
       -r regions.inp
       -o Output.xyz
       -l Logfile.log
```

where `Nprocs` is the number of MPI cores that are going to be used by LICHEM, and `NCpus` is the number of CPUs used by QM and MM wrappers. The MPI nodes are defined by the argument `--hostfile` which includes the names of the available nodes for MPI execution.

2.1.1 Load Balancing with `lichem.MPI`

The number of MPI nodes (`Nprocs`) that is used for hybrid–parallel execution is important for load balancing so that the number of idle MPI nodes is minimized. In QSM calculations, the reactant bead is always owned by the first processor, while the product bead is always owned by the last processor. The remaining beads are then distributed in a cyclic manner;

Processor ID	**Bead ID**			
P_0	0	1	$Nproc+2$	$\cdots$
P_1	2	$Nproc+3$	$\cdots$	
P_2	3	$Nproc+4$	$\cdots$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$P_{Nproc-2}$	$Nproc$	$2(Nproc+1)-1$	$\cdots$	
$P_{Nproc-1}$	$Nproc+1$	$2(Nproc+1)$	$\cdots$	$Nbead-1$

where bead 0 is the reactant and bead $Nbead-1$ is the product. Calculations for reactant and product beads are performed only in the first step since the QSM algorithm requires frozen ends. The best load balance in QSM simulations is achieved when;

$$Nprocs = Nbeads - 2$$

where $Nprocs$ is the total number of MPI nodes, and $Nbeads$ is the total number of beads An example for the distribution of the beads among MPI nodes when the number of beads is larger than the available number of MPI nodes is depicted in Figure 2.1.

2.2 XYZ input files

The input structure for LICHEM is a standard XYZ file with a blank comment line.

N

A X_A Y_A Z_A
B X_B Y_B Z_B
...

Here N is the number of atoms and (X_i,Y_i,Z_i) is the position of particle i. Note that the atom types given in the XYZ file need to be the atomic symbols from the periodic table, not the MM atom types. Additionally, LICHEM reads the XYZ file item-by-item, which means that having additional values on a line will cause (silent) errors.

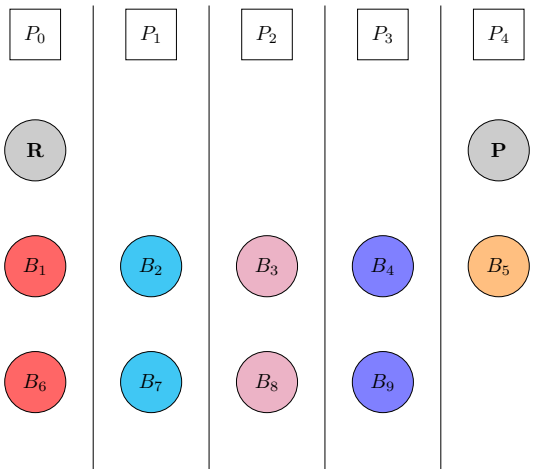


Figure 2.1: An example representation for distribution of 11 beads (B_i, colered circles) among 5 processors (P_i, squares). This distribution assumes full OpenMP–parallel utilization for each bead. Frozen ends are shown with gray circles. Reactant bead is depicted with **R** while the product bead is depicted with **P**.

2.3 Connectivity input files

The connectivity of the molecules and the force field information are defined in the connectivity file. The general format is given below.

id MMTyp NumTyp q Nbonds [connectivity]
...

Here id is the index of the atom (0 to N-1), MMTyp is the force field atom type (i.e. AMBER atom types), NumTyp is a numerical atom type (i.e. TINKER atom types), q is the force field charge on the atom, Nbonds is the number of bonds to the atom, and [connectivity] is the bond list. The length of the connectivity list must match Nbonds and the information in the connectivity file must be given for all atoms, even if there are no MM atoms in the system. During pure QM simulations, most of the connectivity information is ignored. An additional caveat is that the atoms must be listed in order. This is part of an internal check to make sure correct input files were provided.

2.4 Region input files

The region file contains QM/MM regions (QM atoms, pseudobond atoms, boundary atoms, and frozen atoms) as well as input keywords needed to define the type of calculations that will be performed. Blank templates of region files are provided in the documentation directory.

2.4.1 Keywords

Calculations types & wrappers

calculation_type : Type of calculation that will be performed. Calculation types are documented in the Section 2.5 and Chapter 3. Current options are;

- energy
- opt
- dfp
- sd
- neb
- qsm
- fbneb
- pimc

Some additional synonyms are accepted for each type of calculation, but not all synonyms are well documented. Default: N/A

mm_type : Name of the MM wrapper. Current wrappers: TINKER or LAMMPS. Default: N/A

potential_type : Type of interaction potential (QM or MM or QM/MM). This keyword is required and has no default. Default: N/A

qm_type : Name of the QM wrapper. Current wrappers: Gaussian or PSI4 or NWChem. When Gaussian is the QM wrapper, two options are available. If Gaussian09 is wanted to be used then keyword option can be either Gaussian or g09. If Gaussian16 is wanted to be used then keyword option should be g16. Default: N/A

MM Calculations

box_size : Three lattice constants (Å) for the periodic simulation box. Default: 10000.0 10000.0 10000.0

electrostatics : Type of MM electrostatic potential. Current options: Charges or AMOEBA. Default: N/A

ensemble : Thermodynamic ensemble (NVT or NPT). Default: N/A

force_constant : Initial force constant to restrain positions in MM calculations. Default is 100.0

lrec_cut : LREC cutoff (Å) for the smoothing function. This keyword also sets the MM cutoff for energy calculations. Default: 1000.0

lrec_exponent : Integer exponent for the LREC smoothing function. Default: 3

mm_opt_cut : Value of the MM optimization cutoff (Å). This keyword overrides the LREC_Cut keyword during the MM optimizations. Default: 1000.0

mm_opt_tolerance : RMS deviation criteria for stoping the MM and QM/MM optimizations (Å). Force tolerances are generated automatically based on this value. Default: 1e-2

pbc : Use periodic boundary conditions (Yes/No). Default: No

pressure : Pressure of the simulation (atm). Default: 0.0

restrain_mm : String (yes/**no**). Keyword to indicate if the positions of the atoms in MM region are going to be restrained in initial QMMM optimization steps (pre-iterations). The number of pre-iterations are defined by the value of force constant which is divided by 2 in every step until it is < 2. For instance, if the initial force constant is given as 100.0, then the pre-iterations will involve 6 QMMM steps with restrains on MM atoms, and 1 QMMM step without restrains;

1. QMMM step, F=100.0
2. QMMM step, F=50.0
3. QMMM step, F=25.0
4. QMMM step, F=12.5
5. QMMM step, F=6.25

6. QMMM step, F=3.125

7. QMMM step, without restrain

Currently, restrained MM calculations can only be performed with TINKER and during QSM calculations.

temperature : Temperature of the simulation (Kelvin). Default: 300.0

use_ewald : Use Ewald summation for the MM calculations (Yes/No). This keyword turns off MM cutoffs for energy calculations. Default: No

use_lrec : Use a long-range smoothing function (Yes/No) for QM periodic boundary conditions (Yes/No). Default: No

use_mm_cutoff : Use a MM cutoff in optimizations (Yes/No). This can greatly speed up the optimizations, but may prevent the optimization from locating a true minimum. This cutoff only effects the energies and forces in the MM optimizations. Using PBC and LREC is preferred since QM and QM/MM energies are not calculated using an electrostatic cutoff. Default: No

use_solvent : Use an implicit solvent for MM calculations (Yes/No). Implicit solvents can only be used for non-periodic simulations. Using an implicit solvent disables MM cutoffs in energy calculations. Default: No

Monte Carlo Simulations

acceptance_ratio : Fraction of Monte Carlo moves that will be accepted. LICHEM will adjust the Monte Carlo stepsize during the equilibration run to maintain the specified acceptance ratio. Default: 0.50

eq_steps : Number of Monte Carlo and molecular dynamics equilibration steps. Default: 0

prod_steps : Number of Monte Carlo and molecular dynamics steps for production runs. Default: 0

Path Optimizations

beads : Number of replica beads for reaction pathways and path-integral Monte Carlo simulations. Default: 1

frozen_ends : Freeze the end points of the reaction path optimizations (Yes/No). For QSM simulations, this keyword should be set to Yes. Default: No

init_path_chk : Copy the checkpoints from the previous point on the path (Yes/No). Using this option can accelerate the early stages of reaction path calculations by starting some of the points with a checkpoint guess. This keyword only has an effect on the initial calculation of the energies. Default: Yes

max_opt_steps : Maximum number of steps for a QM/MM optimization. Default: 200

max_stepsize : Maximum displacement during an optimization step (Å). Default: 0.10

max_qm_steps : Integer. Maximum number of steps for QM optimization. It indicates that each QM/MM optimization step can involve "*max_qm_steps*" at most;

$$N \text{ QMMM step} \quad (N \leq \text{max\_opt\_steps}) \rightarrow \begin{cases} \text{i.} & n \text{ QM steps} \quad (n \leq max\_qm\_steps) \\ \text{ii.} & 1 \text{ MM step} \end{cases}$$

opt_stepsize : Initial scale factor for the geometry optimizers. A value of zero does nothing and a value of 1.0 takes uses a stepsize based on the forces and/or Hessian. The LICHEM optimizers will attempt to adjust the stepsize to improve performance. Default: 1.0

spring_constant : Nudged elastic band spring constant (eV/Å^2). Default: 1.0

ts_freqs : Automatically calculate frequencies for the optimized climbing image nudged elastic band transition states. Default: No

QM Calculations

qm_basis : Basis set for the QM calculations. If the QM method is SemiEmp, then this keyword should be a model Hamiltonian. Default: N/A

qm_charge : Charge on the QM region. Default: 0

qm_max_force_tol : Real, positive. Use for QM calculations during path optimizations. Tolerance for maximum force among forces of the QM atoms along the path in Hartree/Bohr. Default value is 1e-2 Hartree/Bohr.

qm_memory : Amount of RAM for the QM calculations. This keyword requires two peices of input. An integer input for the RAM and a string for the units (MB or GB). Default: 256 MB

qm_method : QM level of theory (HF or functional name or SemiEmp). If the QM method is set to SemiEmp, then the QM basis should be set to the semi-empirical model Hamiltonian. Default: N/A

qm_opt_tolerance : RMS deviation criteria for stoping the QM optimizations. Use for QM calculations during path optimizations. (Å).

qm_rms_force_tol	:	Real, positive. Tolerance for RMS forces of the QM atoms along the path in Hartree/Bohr. Default value is 5e-3 Hartree/Bohr. Use for QM calculations during path optimizations.
qm_spin	:	Multiplicity of the QM region. Default: 1
qm_units	:	Sets the units used inside the QM wrappers (e.g. Bohr). LICHEM input still needs to be given in Angstrom units. Default: Angstrom
solv_model	:	Type of implicit solvation model. As a general note, solvation models should be chosen carefully. Default: N/A

QM/MM regions

qm_atoms	:	Nqm [list of ids]
pseudobond_atoms	:	Npseudo [list of ids]
boundary_atoms	:	Nbound [list of ids]
frozen_atoms	:	Ninact [list of ids]
neb_atoms	:	Nneb [list of ids]

Definitions of the QM, pseudobond atoms, and boundary atoms can be found in Chapter 3. Frozen atoms are MM atoms that should remain stationary during optimizations, dynamics, or Monte Carlo simulations. Frozen atoms are useful for optimizing small regions of large periodic systems. NEB atoms are the atoms included in the definition of the nudged-elastic band tangents. If no NEB atoms are defined, all QM pseudobond atoms are used to define the tangents. In case of the QSM, active atoms (defined by `neb_atoms`) are used in tangent calculation, and for spacing out when necessary. Any atoms that is not an active atom will be disregarded during the tangent calculations and spacing out.

Note that in pure QM and pure MM simulations, {Nqm,Npseudo,Nbound} can all be set to zero. This is because these regions are not relevant unless LICHEM is performing a QM/MM calculation. For all of the regions, the lists of atoms can be separated by either spaces or newlines.

Debugging & printing

debug : String (yes/no). Keyword for debugging. If it is set to *yes*, four different text files (.txt) are kept per QM optimization step;

- p0file_N.txt : Coordinates of QM and pseudobond atoms within all beads at the begining of N^{th} optimization step.
- grad_N_n.txt : Gradients (xyz components) of each QM and pseudobond atom within all beads for N^{th} optimization step, and n^{th} QM step.
- Initialpath_N_n.txt : Coordinates of QM and pseudobond atoms within all beads before each integration step for N^{th} optimization step, and n^{th} QM step.
- Wholepath_N_n.txt : Coordinates of QM and pseudobond atoms within all beads after each integration step at N^{th} optimization step, and n^{th} QM step.
- Hessian_N_n.txt : Global Hessian that involves Hessians of all beads at N^{th} optimization step, and n^{th} QM step.

All of the text files are kept under the corresponding debug_N directory where N is the optimization step. If it is set to *yes*, then keep_files is set to *yes* and outputs from QM and MM wrappers are saved for the first and the last optimization steps. Default option is *no*.

keep_files : String (yes/no). LICHEM does not keep outputs from QM and MM wrappers by default. However, all outputs from QM and MM wrappers can be kept by setting this keyword to *yes*. If it is set to *yes*, outputs from first and the last optimization steps are always kept.

per_opt_step : Integer. This keyword is used if `keep_files` option is set to yes. Outputs from QM and MM wrappers are stored under the directory named as LICHM_QSM_Opt_N per optimization step defined by this keyword ("N" indicates the optimization step). Each LICHM_QSM_Opt_N directory involve the subdirectories

- LICHM_MM: Outputs from MM wrappers for the corresponding optimization step.
- LICHM_QM_n: Outputs from QM wrappers (see `per_qm_step`) for the corresponding optimization step

Default value is the maximum number of optimization steps.

per_qm_step	:	Integer. This keyword is used if `keep_files` option is set to yes. Outputs from QM wrappers are stored under the directory named as LICHM_QM_n per QM optimization step defined by this keyword ("n" indicates the QM step). Default value is the maximum number of QM steps.
print_normal_modes	:	Causes LICHEM to print normal modes for pure QM frequency calculations and imaginary frequencies for QM/MM frequency calculations (Yes/No). Default: No
print_steps	:	Print Monte Carlo and molecular dynamics data every N steps. Default: 5000

2.5 Calculation types and synonyms

2.5.1 Single-point energies

Single-point energies can be calculated for QM, MM, and QM/MM potentials. Synonyms: Single-point, Energy, SP

2.5.2 Optimizations

Geometry optimizations can be performed with a couple different algorithms.

Native wrapper optimizers: Perform a geometry optimization with the QM or MM optimizers. This option is available for pure QM and MM optimizations. Synonyms: Optimize, opt

Steepest descent: LICHEM has steepest descent optimizer [6] for the QM atoms. The stepsize (recommended: 0.0125) is adjusted to improve convergence. When using this optimizer, the MM regions are optimized with the MM wrapper using a modified force field. Synonyms: Steep, SD

Davidon-Fletcher-Powell: LICHEM has DFP optimizer [6, 7] for the QM atoms. The DFP algorithm is similar to the BGFS method. The stepsize (recommended: 1.0) is adjusted to improve convergence. When using this optimizer, the MM regions are optimized with the MM wrapper using a modified force field. Synonyms: DFP

2.5.3 Reaction Paths and Restrained MM Optimization

Two "chain–of–replica" algorithms are available for reaction path optimization in LICHEM climbing image nudged elastic band (CI–NEB) [8, 9] and quadratic string method (QSM) [10] in both methods the path is represented by a string of beads that are optimized in parallel to the minimum energy path (MEP).
For QSM, LICHEM also has the ability to perform a **Restrained MM Optimization**, where the MM environment of every bead is optimized in a series of iterations with diminishing restraints to reduce the possibility of path discontinuities (see Figure 2.2) [11, 12]
In the NEB algorithm, the path is optimized in parallel using only energies and forces. The images are separated along the reaction path using an additional spring force. The forces are modified such that the

transition state moves up hill. A global DFP optimizer is to used to update the positions of the entire path simultaneously. The stepsize (recommended: 1.0) is adjusted to improve convergence.
The QSM algorithms is designed to be used with frozen ends; only the beads between reactant and product bead are optimized. The flowchart for the QSM algorithm as implemented in LICHEM, with or without restrained MM environment, is given in Figure 2.2.

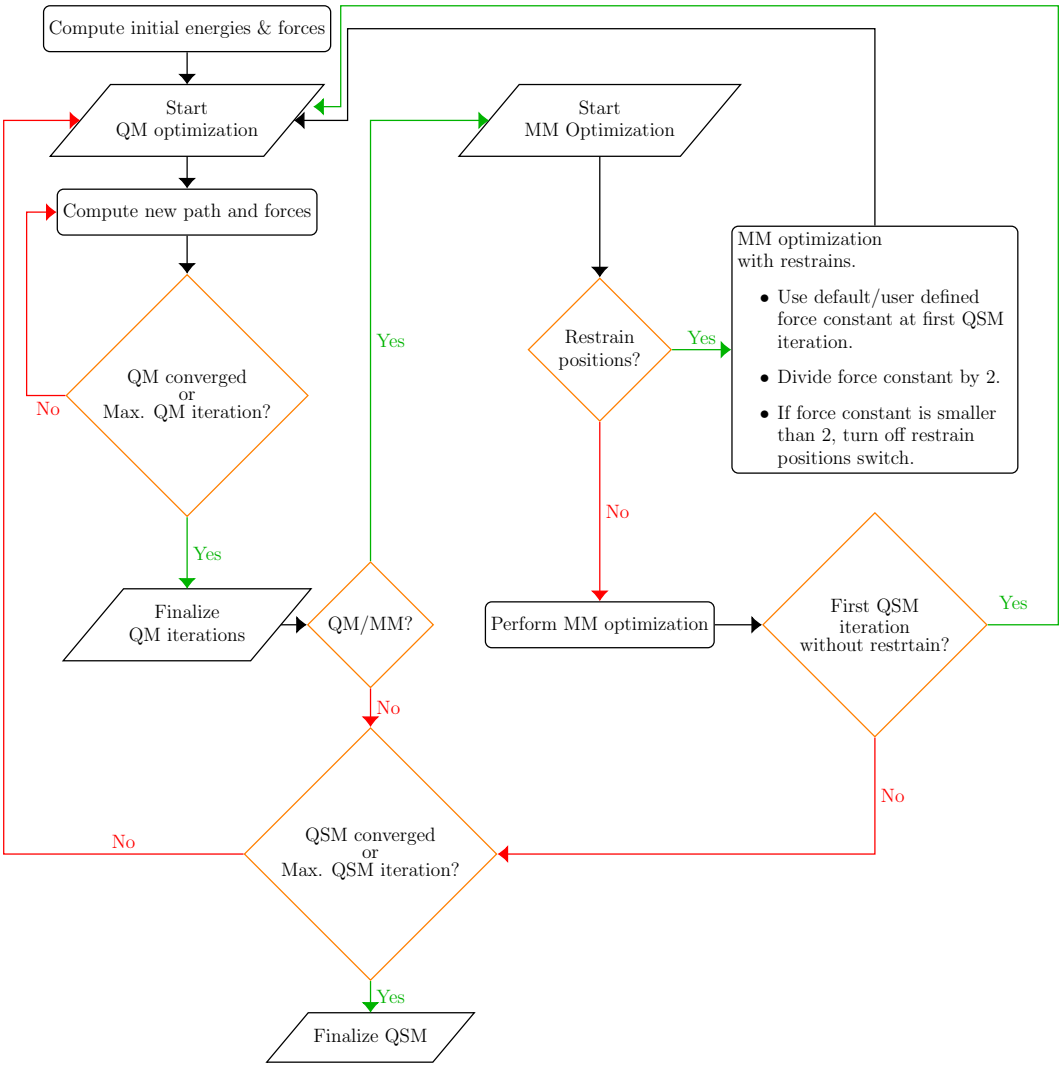


Figure 2.2: QSM algorithm that is implemented within LICHEM.

NEB methods are insensitive to the spring constant (recommended: 1.0). When using this optimizer, the MM regions are optimized with the MM wrapper using a modified force field. Synonyms: NEB, CINEB, CI-NEB

2.5.4 Ensemble sampling

Path-integral Monte Carlo: LICHEM can perform Monte Carlo or path-integral Monte Carlo simulations. The MC stepsize is adjusted during the equilibration step to maintain the specified acceptance ratio.

Synonyms: PIMC

2.6 File conversion

While LICHEM does not handle atom typing or structure generation, it is possible to use the native input/output functions to create LICHEM input from input used for QM and MM packages. In all cases the converter produces three files: xyzfile.xyz connect.inp regions.inp

General QM input

For pure QM calculations, LICHEM can create connectivity files using data from the periodic table.

```
user:$ lichem -convert -q xyzfile.xyz
```

LICHEM can also use a region file to create a BASIS set file for the Gaussian and NWChem basis set input. A BASIS file is required when mixed basis sets or pseudobonds are included in the QM/MM calculations.

```
user:$ lichem -convert -b regions.inp
```

The `-convert` flag checks for specific keywords in the region file to construct a BASIS file in the correct order.

Required keywords:
'QM_type:'
'QM_basis:'
'QM_atoms:'
'Pseudobond_atoms:'

Currently, these keywords must be spelled and formatted exactly as given above.

Multi-replica trajectories:

LICHEM can create or split merged trajectory files for reaction path simulations.

```
user:$ lichem -path -b Nbeads -r Reactant.xyz -t TS.xyz -p Product.xyz
```

The `-path` flag will linearly interpolate between reactant and product structures to create an initial reaction pathway (BeadStartStruct.xyz) for LICHEM. Adding a guess of the transition state structure using the `-t` flag is optional, but can greatly improve the initial path.

```
user:$ lichem -splitpath -b Nbeads -f FrameID -p Path.xyz
```

The `-splitpath` flag will parse a LICHEM multi-replica output or restart file to create a new trajectory file with one replica per frame. Here FrameID is the frame of the merged trajectory (Path.xyz) printed

by the multi-replica simulation.

TINKER:

The file converter for TINKER can be used on standard TINKER XYZ files, TINKER QM/MM XYZ files (g09MM2,Q-CHEM), and TINKER XYZ files with periodic boundary conditions.

```
user:$ lichem -convert -t TINKER.xyz -k tinker.key ( -p Yes)
```

The -p flag is optional and tells LICHEM to read the lattice constants from the second line of the XYZ file.

LICHEM can also be used to create TINKER xyz files:

```
user:$ lichem -tinker -x xyzfile.xyz -c connectfile.inp
```

A file called "tinkxyz.xyz" will be produced from this conversion.

The -GlobalPoles flag reads the force field and initial structure, then prints the multipoles in the global reference frame.

```
user:$ lichem -GlobalPoles -n Ncpus -x xyzfile.xyz -r regions.inp -c connect.inp
```

LAMMPS:

Still in development

2.7 LICHEM output

Most of the LICHEM output is sent through the c++ std output stream. However, trajectories and optimized structures are printed into the file defined by the -o flag. In single-point energy calculations, the output XYZ file is deleted after the calculation is completed, since the geometry does not change during the calculation. LICHEM also leaves some restart files in theworking directory (e.g. Gaussian checkpoint files, multi-replica XYZ files, etc).

2.8 Additional input for wrappers

The LICHEM wrappers automatically look for a couple "extra" input files. These files depend on which wrappers are used in the calculation, however, in most cases the files contain force field and pseudopotential information. Some of the extra files are detailed below in no particular order and examples can be found in the documentation directory.

2.8.1 TINKER

The TINKER wrapper checks the working directory for a file called tinker.key and will read additional force field information (e.g. atom classes, multipoles, etc) based on the keywords listed in that file. Any

keyword defined in tinker.key will be passed into the TINKER key file used by the wrapper. For obvious reasons, the user must also provide the force field parameter files defined in the tinker.key file.

2.8.2 LAMMPS

Still in development

2.8.3 Gaussian

The Gaussian wrapper will look for checkpoint files and a file called BASIS in the working directory. The BASIS file contains basis set and pseudopotential information for QM/MM calculations with bonds between the QM and MM regions (see below). While the file is read in automatically if it is found, the user still needs to tell Gaussian to use this information:

QM_basis: GEN

GEN is a Gaussian keyword which causes the package to check for extra basis set information below the structure.

LICHEM automatically tells Gaussian to look for pseudopotentials, however, these must be given below the basis set information in the BASIS file. Both the basis set and pseudopotential information must be given in Gaussian format.

2.8.4 NWChem

The NWChem wrapper will look for checkpoint files and a file called BASIS in the working directory. The BASIS file contains basis set and pseudopotential information for QM/MM calculations with bonds between the QM and MM regions (see below). If the BASIS file is found, then the basis set given in the regions file is ignored. Both the basis set and pseudopotential information must be given in NWChem format.

2.8.5 Restart files

While restarting incomplete geometry optimizations only requires changing the initial structure given with the -x flag, multi-replica simulations require structural information for each replica. When a multi-replica (PIMC or reaction path) simulation is started LICHEM checks the working directory for a restart file named "BeadStartStruct.xyz" which will be used as the initial structure. The restart file is similar to a standard XYZ file, except all the replicas are merged together (see below).

M

A $X_{A,0}$ $Y_{A,0}$ $Z_{A,0}$
A $X_{A,1}$ $Y_{A,1}$ $Z_{A,1}$
...
A $X_{A,M-1}$ $Y_{A,M-1}$ $Z_{A,M-1}$
B $X_{B,0}$ $Y_{B,0}$ $Z_{B,0}$
B $X_{B,1}$ $Y_{B,1}$ $Z_{B,1}$

...

Here M is the number of atoms times the number of beads, and ($X_{i,j}$,$Y_{i,j}$,$Z_{i,j}$) is the position of particle i in replica j. Note that the atom types given in the XYZ file are ignored since this information is read from the files given by the `-x` and `-c` flags.

2.8.6 Examples

Example LICHEM input and tutorials can be found in the doc directory. Most of the examples are not intended to be used for calculations.

Gaussian_files: Example files for Gaussian basis and pseudopotential input (e.g. BASIS).

PDB_input: A tutorial for creating LICHEM (Gaussian/TINKER) input from a PDB file. The PDB tutorial contains complete input to run a single-point energy calculation on a peptide chain.

Region_files: Example region files for a variety of calculations.

TINKER_files: Contains tinker.key files.

Additional examples can be found in the tests directory.

Chapter 3

Theoretical Background

3.1 Preface

The following chapter is intended to serve as a general introduction to computational chemistry and the methods used in LICHEM. There are numerous places users can find more detailed and up-to-date discussions of these topics. This chapter is only intended to give a broad review of the concepts and terminology.

3.2 Classical methods

3.2.1 Newton's laws of motion

Much of classical physics can be described in terms of Newton's three laws of motion.

1) An object in motion tends to stay in motion unless it is acted upon by a force.
2) The force on an object is equal to its mass times its acceleration ($F = ma$).
3) For every action, there is an equal and opposite reaction.

Newton's second law can be employed in calculations to predict the position and velocity of objects moving in a potential field. A simplified algorithm for the discretized equations of motion can be written as

$$F_n = -\nabla V_n = m\ddot{\mathbf{r}}_n \ , \tag{3.1}$$

$$\dot{\mathbf{r}}_{n+1} = \dot{\mathbf{r}}_n + \ddot{\mathbf{r}}_n \Delta t \ , \tag{3.2}$$

and

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \dot{\mathbf{r}}_n \Delta t + \frac{\ddot{\mathbf{r}}_n \Delta t^2}{2} \ , \tag{3.3}$$

where F is the force, V is the potential energy, $\mathbf{r}$ is a vector representing the positions of all atoms in the system, and Δt is the time interval between step n and $n+1$. The natural ensemble for molecular dynamics is the microcanonical ensemble (NVE), where the number of particles and energy are conserved inside a constant volume simulation box. Although the NVE simulations are the native result of propagating Newton's equations, most experiments are performed in the canonical (NVT) or isothermal-isobaric (NPT) ensembles. In the NVT ensemble, the temperature is held constant instead of the energy, and the NPT ensemble also holds the pressure to be constant by adjusting the size of the simulation box.

3.2.2 Force fields

Empirical force fields are classical potentials which can estimate the potential energy of a system. Although any functional form could be employed for the force field, the potentials are often derived from experiments, classical physics, or quantum theory.

Non-bonded interactions of atoms are typically modeled using point-charges and Lennard-Jones potentials or other van der Waals potentials. The non-bonded potential is given by

$$V(R) = V_{coul}(R) + V_{vdW}(R) \ , \tag{3.4}$$

$$V_{coul}(R) = C \sum \frac{q_i q_j}{r_{ij}} \ , \tag{3.5}$$

and

$$V_{vdW}(R) = \sum \frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} \ , \tag{3.6}$$

where R is a vector representing the coordinates of the system, $V(R)$ is the potential, $V_{coul}(R)$ is the Coulomb potential for point-charges (q), C is a constant, r_{ij} is the distance between atoms i and j, $V_{vdW}(R)$ is the van der Waals potential (in this case, the Lennard-Jones potential), A_{ij} is a parameter representing the exchange interaction, and B_{ij} is a parameter representing the dispersion interaction.

Bonded interactions are often represented by the harmonic potential. Harmonic bonds behave as idealized springs that never weaken or break. The harmonic potential is given by

$$V_{harm}(r_{ij}) = \frac{K_{ij}}{2} r_{ij}^2 \ , \tag{3.7}$$

where K_{ij} is the spring constant. Similarly, bond angles can be described by a harmonic potential

$$V_{harm}(\theta_{ijk}) = \frac{K_{ijk}}{2} \theta_{ijk}^2 \ , \tag{3.8}$$

where θ is the angle between atoms i, j, and k. More complicated potentials can be constructed for the bonded interactions, but these are two of the most common potentials. Typically, large molecules require potentials for dihedral angles and improper dihedral angles.

The parameters for force fields are determined in a very systematic fashion. Typically, the charges are determined from electronic structure calculations and bonded parameters are determined by matching the molecular vibrational modes. The van der Waals parameters are usually adjusted to reproduce the density, enthalpy of freezing, or other experimentally determined properties.

3.2.3 Multipolar/polarizable force fields

The electron density and nuclei in a molecule create a continuous, and often, anisotropic electrostatic potential. In classical simulations, interactions due to the diffuse electron density are generally approximated by a set of pair-wise electrostatic potentials. One of the simplest methods for modeling the molecular electrostatic field is to place point–charges (monopoles) on the atomic centers. Atomic monopoles are often inadequate [13, 14], and some models augment the field by adding massless charged dummy atoms to the molecule [15, 16]. However, the dummy atoms can complicate the potential/dynamics during

molecular simulations.

An alternative approach is to add higher–order moments (multipoles) of the electrostatic potential to the model [13, 17, 18, 19]. A multipole expansion is a Taylor series representation of the continuous electrostatic potential (V_{elst}) [19],

$$V_{elst}(r) \approx \sum_{n}^{\infty} \frac{d}{n!\, dr^n} V_{elst}(r) r^n \ , \tag{3.9}$$

where r is the distance from the point of interest (e.g. nucleus, center of mass, etc) and n is the order of the moment. Typically, MM force fields truncate the Taylor series at the second order [13, 17, 18], and the moments can be determined from an analysis of the electron density or electrostatic potential.

The zeroth moment of the electrostatic potential is the monopole, which is equivalent to a point–charge. The first and second moments are referred to as the dipole and quadrupole, respectively. Multipole moments can produce an anisotropic electrostatic potential around individual atoms, as opposed to the spherical potential wells produced by monopoles. Thus, a set of atomic multipole moments, through the quadrupole moment, can reproduce the molecular electrostatic potential with a higher degree of accuracy than monopole moments alone [13, 19]. Atom centered multipolar models provide a reasonable compromise between computational cost and accuracy in the reproduction of the electrostatic potential at medium and long–range.

3.2.4 Polarizable density force fields

Polarizable density force fields are a special class of polarizable force fields where the electron density is represented by Gaussian functions. These potentials can greatly improve the calculations of the electrostatic field and can include exchange and charge penetration effects.

3.3 Geometry and reaction path optimizations

3.3.1 Steepest descent

Steepest descent (SD) is the simplest optimization method. The gradient of the potential is calculated at the current geometry and used to update the atomic coordinates.

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \alpha_n \nabla V(\mathbf{r}) \ , \tag{3.10}$$

Here $V(\mathbf{r})$ is the potential at the set of coordinates $\mathbf{r}$ and α is a scale factor which determines how far a particle moves during step n.

Efficient SD optimizers use line search algorithms which backtrack to find the optimum value of α at step n. However, SD optimizations often do not converge even with optimal step sizes.

The SD algorithm in LICHEM does not use a line search to optimize α at each step. Instead, a maximum value of α is given in the input and the optimization starts with $\alpha = 0.7\alpha_{max}$. If the QM/MM energy is decreasing, then the step size increases by 5% until it reaches α_{max}. If the energy increases, the step size is reduced by 40% and the optimization continues.

3.3.2 Davidon-Fletcher-Powell

SD optimizations move along the first derivative of the potential (gradient). Optimzations are more efficient if the atoms move along a path chosen using both the first and second derivatives of the potential. The second derivative matrix (Hessian) can be expensive to calculate, however, there are several quasi-Newton algorithms which can improve a guess Hessian. The most common procedures are the Davidon-Fletcher-Powell (DFP) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) algorithms. These two methods are mathematically related and can theoretically start from any initial guess of the inverse Hessian, H^{-1}.

The DFP algorithm can update the inverse Hessian using the forces (F_n), the current inverse Hessian, and the optimization vector (p_n). The optimization algorithm is given by

$$p_n = \alpha_n H_n^{-1} F_n \ , \tag{3.11}$$

$$\delta_n = F_{n-1} - F_n \ , \tag{3.12}$$

and

$$H_{n+1}^{-1} = H_n^{-1} + \frac{p_n p_n^T}{p_n^T \delta_n} - \frac{H_n^{-1} \delta_n \delta_n^T H_n^{-1}}{\delta_n^T H_n^{-1} \delta_n} \ , \tag{3.13}$$

where δ is the change in the forces. The BFGS algorithm is similar to the DFP algorithm and can be obtained by replacing the inverse Hessian in Eq. 3.13 with the Hessian.

If the updates satisfy the Wolfe conditions, then the algorithm will converge to the correct solution. However, satisfying the Wolfe conditions requires a line search algorithm and storing old structures.

To keep the LICHEM DFP optimizer simple, the guess inverse Hessian is chosen to be the identity matrix, which effectively causes the first optimization step to be a SD step. As with the SD algorithm, the step size is adjusted based on the progress of the optimizer. Initially, $\alpha = 0.02\alpha_{max}$ and the step size is increased by 20% after each sucessful optimization step. If the energy does not decrease or if 30 optimization steps have been performed, the Hessian is reset to the identity matrix and $\alpha = 0.02\alpha_{max}$. This algorithm will not always satisfy the Wolfe conditions, however, it will not violate the conditions for long before a new Hessian is constructed.

3.3.3 Nudged elastic band

Algorithms for geometry optimizations can only locate stationary points (e.g. minima and transition states). An entire reaction path can be optimized by performing an optimization with multiple replicas between the reactant and product states. Unfortunately, all of the replicas would optimize to either a minimum or a transition state.

In order to sample regions between the stationary points, spring forces can be added to the forces. The modified elastic band forces (F') on bead p and step n are given by

$$F'_{p,n} = F_{p,n} + K\Delta R_{p,n} \ , \tag{3.14}$$

and

$$\Delta R_{p,n} = |R_{p+1,n} - R_{p,n}| - |R_{p,n} - R_{p-1,n}| \ , \tag{3.15}$$

where K is the spring constant, and ΔR is the displacement between replicas. The spring forces help the elastic band method locate the minimum energy path, however, the beads still want to slide down to the minima.

The forces can be further modified by subtracting the forces parallel to the reaction path. The forces are then given by [8, 9]

$$F'_{p,n} = F_{p,n} - F_{p,n} \cdot \tau_{p,n}\tau_{p,n} + K\Delta R_{p,n}\tau_{p,n} \ , \tag{3.16}$$

where τ is a unit tanget vector representing the direction of the reaction path. Removing the tangent components of the forces is referred to as nudging in the literature [8, 9].

The nudged elastic band method (NEB) is relatively insensitive to the magnitude of the spring constant. Unfortunately, the NEB method is still unlikely to locate a transition state. The climbing image (CI) NEB method [8] further modifies the forces of the highest energy. The CI travels upwards along the tangent to find a transition state structure. Transition state forces on the CI (replica c) are now given by

$$F'_{c,n} = F_{c,n} - 2F_{c,n} \cdot \tau_{c,n}\tau_{c,n} \ , \tag{3.17}$$

where the forces are inverted along the tangent [8].

The tangent vectors are given by [8]

$$\tau_{p,n} = \tau^{+}_{p,n} \quad , \quad p < c \ , \tag{3.18}$$

$$\tau_{p,n} = \tau^{-}_{p,n} \quad , \quad p > c \ , \tag{3.19}$$

$$\tau_{p,n} = \frac{\tau^{+}_{p,n} + \tau^{-}_{p,n}}{|\tau^{+}_{p,n} + \tau^{-}_{p,n}|} \quad , \quad p = c \ , \tag{3.20}$$

$$\tau^{+}_{p,n} = \frac{R_{p+1,n} - R_{p,n}}{|R_{p+1,n} - R_{p,n}|} \ , \tag{3.21}$$

and

$$\tau^{-}_{p,n} = \frac{R_{p,n} - R_{p-1,n}}{|R_{p,n} - R_{p-1,n}|} \ . \tag{3.22}$$

Here the tangents are only calculated in the direction of the transition state.

The CI-NEB forces can be utilized in essentially any optimization algorithm. Currently, LICHEM employs a global DFP (GDFP) optimization. The GDFP optimizer uses all replicas between the end points to construct and update a single large Hessian matrix. With all of the replicas in the Hessian, the GDFP algorithm minimizes the energy along the entire reaction pathway and includes the NEB spring forces in the curvature. Thus, the beads should smoothly converge to the minimum energy path.

If the CI-NEB optimization is performed without freezing the reactant and product end points, then the reactant and product structures each get their own Hessian matrix. Optimizing the end points separately from the remainder of the path prevents nearly converged end points from influencing the GDFP step size.

3.3.4 Quadradic string method

The method eliminates the need for an artificial spring constant, and does not require a predetermined step size. Instead, the surface and the trust radius of the approximate Hessians are updated at the end of each iteration.

The details of the QSM algorithm can be found in the work of Burger and Yang [10]. In our implementation, the QSM algorithm starts with the calculation of energies and gradients of all beads that are provided by the user. The approximate Hessians (H_k) that were constructed with the gradients in the initial step are then updated with the damped Broyden-Fletcher-Goldfarb-Shanno (DBFGS) algorithm;

$$H^{i+1} = H^i - \frac{H^i \delta^i (\delta^i)^T H^i}{(\delta^i)^T H^i \delta^i} + \frac{r^i (r^i)^T}{(r^i)^T \delta^i} \tag{3.23}$$

where i represents the iteration number. The δ^i and r^i in Eqn. 3.23 are defined as;

$$\delta^i = x^{i+1} - g^i \tag{3.24}$$

$$r_i = \theta^i \gamma^i + (1 - \theta^i) H^i \delta^i \tag{3.25}$$

with γ^i and θ^i as;

$$\gamma^i = g^{i+1} - g^i \tag{3.26}$$

$$\theta^i = \begin{cases} 1, & \text{if} (\delta^i)^T \gamma^i > 0.2 (\delta^i)^T H^i \delta^i \\ \dfrac{0.8 (\delta^i)^T H^i \delta^i}{(\delta^i)^T H^i \delta^i - (\delta^i)^T \gamma^i}, & \text{otherwise} \end{cases} \tag{3.27}$$

The update of H^k is followed by the update of the trust radii using the energy as a merit function;

$$\rho = \frac{E_k^{i+1} - E_k^i}{\frac{1}{2} dx_k^T H_k^i dx_k + dx_k^T g_k^i} \tag{3.28}$$

where k represents the respective bead and ρ is used to compute how close the merit function is to the expected value. After the computation of new trust radii, the path is integrated by using Runge–Kutta explicit 4/5 integration scheme with Cash–Karp parameters. The integration is performed until a bead reaches its trust radius, or the calculation is converged. The beads are redistributed if necessary by using a polynomial interpolation scheme. In our QSM implementation, the projection of the gradients, and the redistribution of the beads are performed by taking only the reacting atoms into consideration (see below). If the reacting atoms are not specified, all atoms are considered as active.

3.4 Quantum methods

3.4.1 Schrödinger equation

Quantum behavior can be described using the Schrödinger wave equation and the Hamiltonian operator, $\hat{H}$,

$$\hat{H} = \hat{T} + \hat{V} \ , \tag{3.29}$$

where $\hat{T}$ is the kinetic energy operator and $\hat{V}$ is the potential energy operator. The Schrödinger equation is given by

$$\hat{H}\psi = E\psi \rightarrow E = \int \psi^* \hat{H}\psi d\tau \ , \tag{3.30}$$

where ψ is the wave function, E is the energy, * denotes the complex conjugate, and $\int d\tau$ is the integral over all space. The Schrödinger equation can also be written in terms of a determinant,

$$|H - ES| = 0 \ , \tag{3.31}$$

where S is the overlap matrix. Another common shorthand is bra-ket notation.

$$\hat{H}|\psi\rangle = E|\psi\rangle \rightarrow E = \langle\psi|\hat{H}|\psi\rangle \ , \tag{3.32}$$

where $\langle i|\hat{O}|j\rangle$ is the integral of operator $\hat{O}$ on state j.

3.4.2 Perturbation theory

If there is a complicated system and we want to solve the Schrödinger equation, a good starting point is often to use a system for which the exact or nearly-exact solution is known. In chemistry, this is often the hydrogen atom, and in solid state physics, it is often the "homogeneous electron gas". For example, helium can be approximated as two electrons interacting with a nucleus, but not with each other (i.e. solving the hydrogen problem twice with a nuclear charge of $+2$).

Mathematically, the Hamiltonian can be divided into the simple parts and the ones we wish we did not have to solve,

$$\hat{H} = \hat{H}_0 + \hat{H}_1 + \hat{H}_2 + \cdots \ , \tag{3.33}$$

where $\hat{H}$ is the full Hamiltonian, $\hat{H}_0$ is the simple (zeroith-order) Hamiltonian that we can solve analytically, and $\hat{H}_i$ (i=1,2...) are the ith-order Hamiltonians for interactions that have been neglected in $\hat{H}_0$. The Schrödinger equation can then solved for the simple system,

$$\hat{H}_0\psi_0 = E_0\psi_0 \ . \tag{3.34}$$

Perturbation theory adds the ith-order Hamiltonians to the zeroth-order energy as the average interactions (based on the probability from the wave function). This is easiest to understand for first-order perturbation theory, however, the higher-order corrections are systematic and straightforward. The perturbed Schrödinger equation is given as

$$\hat{H}\psi = E\psi \rightarrow (\hat{H}_0 + \hat{H}_1)\psi = E\psi \ , \tag{3.35}$$

and

$$E \approx E_0 + E_1 \approx E_0 + \langle\psi_0|\hat{H}_1|\psi_0\rangle \ . \tag{3.36}$$

Note that first-order perturbation theory uses the zeroth-order wave function for the averages. This can be seen as both an advantage and a disadvantage in the calculations. The advantage is that it speeds up the calculation to use a lower-order wave function since the higher order terms have no effect on the solution of the simpler system. The disadvantage is that wave function itself does not respond to the higher-order interactions. For helium with a first-order perturbation, the electrons are too close to the nucleus since there is a strong nuclear attraction with no repulsion due to the second electron.

It should be noted that perturbation theory can correct both the wave function and the energy. However, the corrections to the wave function can be two orders lower than the corrections to the energy. A third-order correction to the energy would require a first-order correction to the wave function.

3.4.3 Variational theorem

Another method for solving the Schrödinger equation involves approximating the wave function for a complex system. If the Hamiltonian for a system is known, but not the wave function, we can often make an educated guess. Since the ground-state of a system is the one with the lowest energy, any guess-wave function, ϕ, must have an energy that is greater-than or equal-to the ground-state energy. If the energy calculated using ϕ is equal-to the true ground-state energy, then ϕ is the true ground-state wave function. If $\hat{H}\phi = E\phi$ predicts an energy lower than the ground-state energy, then the ground-state is not the lowest energy state (impossible by definition). This theorem can be proven mathematically, however, it is conceptually intuitive.

While the variational theorem may not sound useful since a reasonable guess wave function must be provided, we can often employ the properties of linear combinations and the solutions of $\hat{H}_0\psi_0 = E_0\psi_0$. When solving differential equations, any linear combination of the solutions to the equation is also a solution. For example, the 1s ground-state of the hydrogen atom is a solution of the Schrödinger equation for the hydrogen atom. The 2s excited state is another solution. Therefore,

$$\phi = c_1\psi_{1s} + c_2\psi_{2s} \ , \tag{3.37}$$

is also a solution for the hydrogen atom. We call this a linear combination of atomic orbitals (LCAO) wave function.

3.4.4 Basis sets

In principle, any set of functions (a basis set) can be chosen as a guess and then the linear combination coefficients (c_1 and c_2 above) can be adjusted to lower the energy. The more functions that are included in the basis set, the closer the guess-wave function can get to the true wave function of the system. A logical choice would be to use hydrogen-like wave functions for the individual atoms in a molecule as the basis set, however, the integrals are not easy to calculate.

A more practical solution is to choose a large set of functions that we can easily integrate. For instance, Gaussian functions have great properties that simplify integration. The STO-3G basis set provides a good example of how this is done. A Slater-type orbital (STO) is a systematic hydrogen-like orbital for any atomic state. The STO-3G basis set represents the STOs of the atoms as the combination of 3 Gaussian functions. This produces approximate "hydrogen-like" orbitals that have still have the mathematical properties of Gaussians. For the hydrogen molecule, the wave function would be given by

$$\psi \approx c_a\phi_{a,1s} + c_b\phi_{b,1s} \ , \tag{3.38}$$

where $\phi_{a,1s}$ and $\phi_{b,1s}$ are the 1s orbitals on the first and second hydrogen atom, respectively. If ϕ_n is represented by the STO-3G basis set, the wavefunciton is given by

$$\psi \approx c_a(G_1^a + G_2^a + G_3^a) + c_b(G_1^b + G_2^b + G_3^b) \ , \tag{3.39}$$

where G_i^j is the ith Gaussian functions centered on atom j. This approximation produces a lot of Gaussian integrals, but only has two coefficients that can be changed to lower the energy. Hence, the basis set has two relatively easy to integrate orbitals made of three Gaussians each.

Next, let's consider the Pople basis sets which are written as w-xyzG. This notation indicates how many Gaussians are used to represent the atomic orbitals. For now, we can just use the lithium atom as the example for the wave function,

$$\psi = c_1\phi_{1s} + c_2\phi_{2s} + c_3\phi_{2p,x} + c_4\phi_{2p,y} + c_5\phi_{2p,z} \ . \tag{3.40}$$

Lithium has five orbitals and five variational coefficients if we consider ϕ_n to be an STO, but we are going to use more complicated basis sets.

The w in w-xyzG represents the number of Gaussian functions used to represent the core orbitals (filled inner shells) and (x,y,z) represent the number of Gaussian functions in the valence orbitals. For the moment let's ignore the p orbitals (ψ') and consider the 3-21G basis set (w=3,x=2,y=1,z=0) for the s orbitals.

$$\psi' = c_{1w}(G_1 + G_2 + G_3) + c_{2x}(G_4 + G_5) + c_{2y}G_6 \ . \tag{3.41}$$

The first three Gaussian functions represent the 1s orbital and have a single variational coefficient. The second three Gaussians split the valence orbital into a two Gaussian and one Gaussian function with two additional variational parameters. These split valence basis sets are more flexible and perform better compared to the STO valence states. The full 3-21G basis set for lithium has nine variational parameters and a total of fifteen Gaussian functions. The larger 6-311G basis set splits the valence states into three parts (thirteen variational parameters, twenty total Gaussian functions). The number of "parts" representing the valence orbitals is referred to as the zeta level of the basis set. A single zeta basis set has one variational parameter per valence orbital (e.g. STO-3G), a double zeta basis set has two variational parameters per valence orbital (e.g. 6-31G), and a triple zeta basis set has three parameters per valence orbital (e.g. 6-311G).

The Pople basis sets are often used with polarization and/or diffuse Gaussian functions. Polarization functions are additional p, d, f, etc. orbitals on top of the standard functions. The 6-31G(d) basis set adds d orbitals to the heavy atoms (i.e. not hydrogen) and 6-31G(d,p) also adds p orbitals to the hydrogens. This can be made much more complicated (i.e. 6-31G(df,pd)) but the notation is straight-forward. There is also an older notation for the polarization functions using * instead of the orbitals (6-31G*=6-31G(d) and 6-31G**=6-31G(d,p)). Diffuse functions are very large s orbitals (e.g. 5s, 6s) added to the atomic basis sets. The notation is 6-31+G and 6-31++G for adding diffuse functions to the heavy atoms and all atoms respectively. The polarization and diffuse functions are often vital for small molecules, but can be problematic in large systems.

Dunning basis sets have a simpler notation, but are mathematically much more complicated. The notation is cc-pVnZ, where n is the zeta level. The notation stands for "correlation consistent polarized valence n zeta" with n=D,T,Q,5,6,etc. A cc-pVDZ basis set is approximately the same as the 6-31G(d,p) basis set. Diffuse functions are added to all atoms with aug-(augmented), i.e. aug-cc-pVnZ. The Dunning basis sets are designed to systematically converge to the infinite basis set, however, they are much more computationally demanding than the Pople basis sets. The extra computational expense is due to the use of spherical harmonic functions.

3.4.5 Pseudopotentials

Large or complicated systems can be simplified by discarding core electrons and modifying the electron-nuclei interactions for the valence states. A simple (albeit extremely inaccurate) example would be to

combine the core electrons of an atom with the nucleus (e.g. oxygen would have a nuclear charge of six and six valence electrons). The modified potentials are referred to as pseudopotentials. Pseudopotentials are useful for a variety of reasons, they reduce the number of electrons, the least important electrons are removed, basis functions have a difficult time modeling the core region, and the potentials can be designed to include relativistic effects.

Real pseudopotentials do not really change the nuclear charge, but rather add an electrostatic potential with the same spatial distribution as the electrons which are replaced. A further advantage of pseudopotentials, which is employed in QM/MM calculations, is that the potentials can replace entire atoms. For instance, a methyl group can be modeled as a fluorine atom combined with a pseudopotential.

3.4.6 Orbital-free density functional theory

Density functional theory (DFT) is an elegant solution to problems in quantum theory and is distinct from the other formalisms (matrix mechanics (Heisenberg), wave mechanics (Schrödinger), and the path-integral approach (Feynman)). The idea is that you do not need to know the wave function, only the electron density of the system, to calculate the energy. It can be proven that there is a one-to-one relationship between a given electron density and the energy of the system. This greatly simplifies the problem the problem since electrons contribute three degrees of freedom (each) to the Schrödinger equation, while the density, ρ, always has a total of three degrees of freedom.

$$\hat{H}\psi = E\psi \rightarrow E[\rho] = T[\rho] + V[\rho] \; . \tag{3.42}$$

DFT, in its original construction, does not require orbitals or basis sets. The entire problem can be solved as a function of $\rho(x, y, z)$. Unfortunately, the exact functional $E[\rho]$ is unknown and the functional must be approximated. Functionals are often based on the homogeneous electron gas, and the nuclei are treated as an external field which perturbs the electrons. Current functionals are only approximate, however, they often provide a computationally efficient solution for large systems.

The earliest orbital-free DFT functional was the Thomas-Fermi (TF) model,

$$T[\rho(\mathbf{r})] = C_{tf} \int \rho(\mathbf{r})^{5/3} d\mathbf{r} \; , \tag{3.43}$$

$$V[\rho(\mathbf{r})] = \int \frac{\rho(\mathbf{r})\rho(\mathbf{r}')}{2|\mathbf{r} - \mathbf{r}'|} d\mathbf{r} d\mathbf{r}' - \sum \int \frac{\rho(\mathbf{r}) Z_i}{|\mathbf{r} - \mathbf{r}_i|} d\mathbf{r} \; , \tag{3.44}$$

and

$$E[\rho(\mathbf{r})] = T[\rho(\mathbf{r})] + V[\rho(\mathbf{r})] \; . \tag{3.45}$$

Here $\mathbf{r}$ is the position of an electron, Z_i is the nuclear charge of atom i, $\mathbf{r}_i$ is the position of nucleus i, and $\mathbf{r}'$ is the position of the second electron. This model only requires an approximation for the electron density and the positions of the nuclei, and hence, the electron density can be treated as a fluid instead of discrete particles.

In general the electron-electron and electron-nucleus parts of the functional are much more accurate than the kinetic energy functional. In fact, the one of the simplest potential functionals (LDA) is still used, but the simplest kinetic energy functional (TF) cannot even predict chemical bonding. A more accurate approximation for the kinetic energy is discussed below.

3.4.7 Self-consistent field methods

The self-consistent field (SCF) or Hartee-Fock (HF) formalism takes advantage of the topics discussed above to find a simple but relatively accurate solution to the Schrödinger equation. The method iteratively solves the equations using the Fock operator, F,

$$E = \langle\phi|\hat{F}|\phi\rangle \ , \tag{3.46}$$

$$\hat{F} = \hat{H}_0 + \langle\phi|\hat{H}_1|\phi\rangle \ , \tag{3.47}$$

until the variational wave function no-longer changes. The problem converges since the Fock operator depends on the average electron-electron repulsion from the previous iteration. The result is an approximate first-order energy and a first-order wave function.

The HF method can be improved by adding additional perturbations. A second-order correction to the HF energy is referred to as MP2, and up to MP4 are often discussed in the literature. The MP2 energy, E_{mp2}, is given by

$$E_{mp2} = E_{hf} + \langle E_2\rangle \ , \tag{3.48}$$

where E_{hf} is the HF energy and $\langle E_2\rangle$ is the second-order perturbation.

There is a bit of terminology that has thus far been missing from the discussions. The first is "electron correlation", which refers to the interactions and motions of the electrons (i.e. an electron wants to move away from another electron). In the literature, the "correlation energy" is the electron-electron interaction energy not captured by the HF method (second-order and higher perturbations). The SCF formalism is only a mean-field approach, and hence, it gives the average interactions not the true interactions. The second bit of terminology is the "exchange interaction", which is included in $\hat{F}$ through $\hat{H}_1$. The solution of the Schrödinger equation for electrons should be anti-symmetric with respects to the exchange of two electrons. The exchange interaction is added to the $\hat{H}_1$ operator to force the variational wave function to have the correct symmetry.

Hartree-Fock calculations capture the exchange, but miss the higher-order correlation. Density functional theory calculations, on the other hand, capture some of the higher-order correlation, but miss some of the exchange interaction.

3.4.8 Kohn-Sham DFT

Kohn-Sham (KS) DFT is a SCF approach similar to HF theory, however, it does not obey the variational theorem. To overcome the failures of the orbital-free kinetic energy functionals, orbitals are reintroduced to the problem so that the kinetic energy can be calculated "exactly" from a basis set. As with the HF method, the electrons are moving in a mean-field of all other electrons, but are not interacting directly. Since KS-DFT has orbitals, the exchange interaction can also be calculated "exactly" with only a modest increase in computational cost. However, since most functionals include an exchange term, hybrid functionals add in a percentage of the exact (HF level) exchange and remove the same percentage of DFT exchange. Researchers can make entire careers out of developing better functionals (there are currently thousands) and the mathematical structure of the functionals can be quite complicated. However, some of the most common functionals are LDA (based on the electron gas, used in solid state physics), PBE (contains no empirical parameters, used in physics), PBE0 (a hybrid form of PBE with

25% HF exchange, used in physics), and B3LYP (a hybrid functional with three empirical parameters, used heavily by chemists).

3.4.9 Dispersion corrections

KS-DFT does a reasonably good job for describing bonded interactions, metals, geometries, and energies. However, DFT poorly describes van der Waals interactions, such as dispersion. Modern functionals are often dispersion corrected by adding empirical corrections to the nuclei-nuclei interactions. For the most part, the dispersion energy, E_d, takes the form

$$E_d = \sum_{i,j} f(R_{ij}) \frac{C_6}{R_{ij}^6} \ , \tag{3.49}$$

where C_6 is an emprical parameter, $f(R_{ij})$ is a damping function, and R_{ij} is the distance between atoms i and j.

3.5 QM/MM

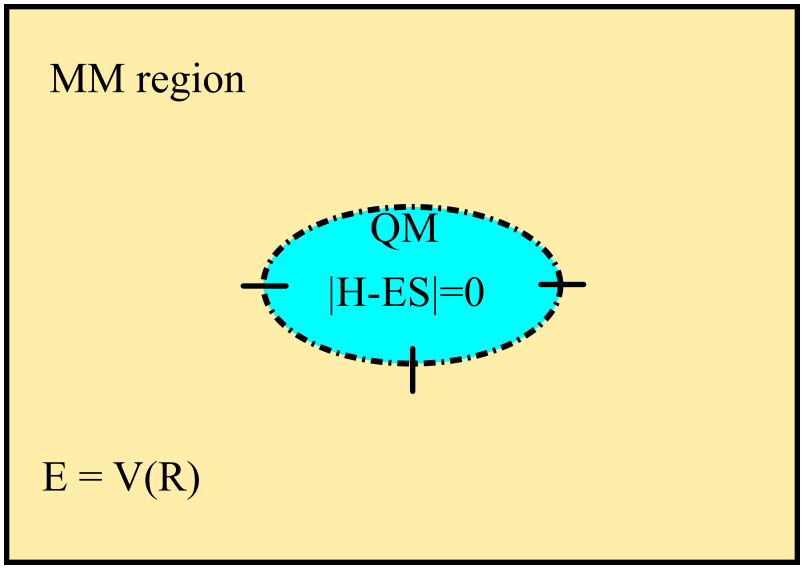


Figure 3.1: Schematic of a QM/MM simulation box with bonds between the QM and MM regions.

QM/MM calculations divide the system into regions which require accurate quantum treatment and regions that can safely be approximated by classical models (see Figure 3.1). The full effective Hamiltonian, $\hat{H}_{eff}$, can be written as

$$\hat{H}_{eff} = \hat{H}_{qm} + \hat{V}_{mm} + \hat{V}_{qmmm} \ , \tag{3.50}$$

where $\hat{H}_{qm}$ is the Hamiltonian for the QM region, $\hat{V}_{mm}$ is the classical potential for the MM region, and $\hat{V}_{qmmm}$ is the potential for the interaction of the QM and MM regions.

Since the MM and QM/MM interface are included only through the potential, solving the Schrödinger equation is essentially trivial.

$$E \approx \langle \psi | \hat{H}_{eff} | \psi \rangle \tag{3.51}$$

3.5.1 QM-MM interactions

When the QM and MM regions are not connected by covalent bonds, the QM-MM interactions are reasonably straightforward. The interaction between the electric field of the MM region (point-charges or multipoles) and the QM region is calculated by adding point-charges to the QM calculations. This allows for electrostatic repulsion and polarization, but neglects exchange and dispersion interactions. These missing interactions are added using the vdW potentials taken from the MM force field.

For a single-point energy, the QM+charges calculation is performed with the QM wrapper, while the MM wrapper calculates the MM energy with no charges or bonded potentials on any of the QM atoms. These two energies are then added together to get the total energy. This approach splits $\hat{V}_{qmmm}$ between the MM calculation ($\hat{V}_{mm}$) and the QM calculation ($\hat{H}_{qm}$).

3.5.2 Pseudobond method

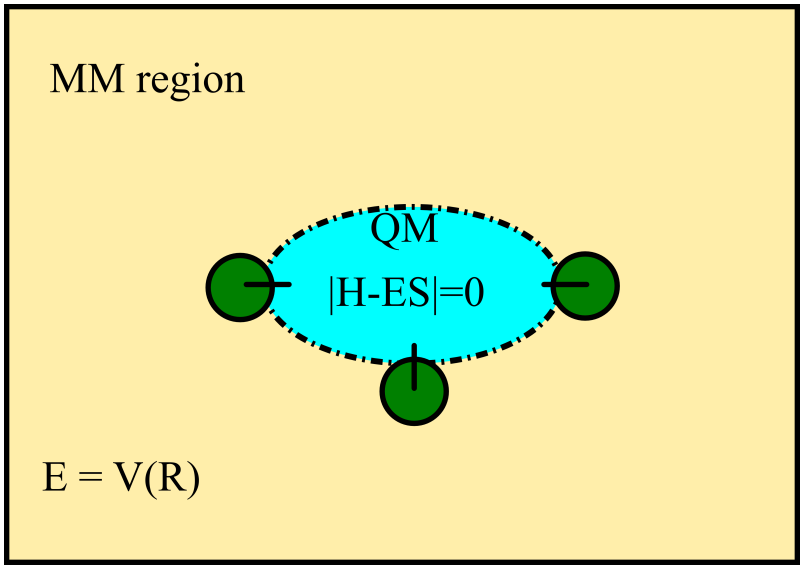


Figure 3.2: Schematic of a QM/MM simulation box with the boundary region shown in green between the QM and MM regions.

The calculations are more complicated when there are bonds between the QM and MM regions. Covalent bonds require two additional regions of the QM/MM system, a pseudobond (PB) region and a boundary atom (BA) region. Pseudo-atoms are shared by the QM and MM regions, and handle the bulk of the covalent interactions. The severed bonds from the MM region are treated by adding a pseudopotential to the pseudo-atoms that allows a fluorine atom to mimic the behavior of an sp^3 hybridized carbon or nitrogen atom. Boundary atoms correct for two additional errors. First, having point-charges close to the QM region can cause the electron density to be over-polarized. The second error is introduced by the fact that the QM region must have an integer number of electrons. Both of these problems can be mitigated by absorbing the point-charges on the atoms bonded to the pseudo-atoms into the QM region during the QM calculation. This produces atoms with zero-charge on the boundaries between the QM and MM regions.

For the MM part of the calculation, the pseudo-atoms and boundary atoms retain their force field charges, and the MM bonding interactions are added for the pseudo-atoms. The treatment of different interaction

	Wrapper calculation	
Int. type	MM	QM
MM-MM	FF	Zero
QM-QM	Zero	ρ_{el}
PA-PA	FF	PP+ρ_{el}
BA-BA	FF	Zero
QM-MM	FF-$\{q_{qm}\}$	ρ_{el}+$\{q_{mm}\}$
QM-PA	Bonds	PP+ρ_{el}
QM-BA	FF-$\{q_{qm}\}$	Zero
MM-PA	FF	PP+ρ_{el}+$\{q_{mm}\}$
MM-BA	FF	Zero
PA-BA	FF	Zero

Table 3.1: The treatment of region-region interactions in the MM and QM wrappers during single-point energy calculations. Interactions were abbreviated as follows: force field (FF), electron density (ρ_{el}), pseudopotential (PP), no interaction (Zero), QM point-charges ($\{q_{qm}\}$), MM bond potentials (Bonds), and MM point-charges ($\{q_{mm}\}$). A "-" sign is used if the interaction is removed instead of added (i.e. FF-$\{q_{qm}\}$ denotes the force field with no charges on the QM atoms).

types are given in Table 3.1 for single-point energy calculations. In the pseudobond approach, only some of the bonded interactions are calculated for the QM/MM interface. Further details on which bonded interactions are included can be found in the literature.

So far we have discussed systems similar to the one shown in Figure 3.2, where the boundary atoms only surround the covalent bonds. This is due to the primitive treatment of the QM-MM boundaries. A more complicated scheme can be employed where the boundary atoms are represented by a polarizable frozen density force field. The QM and frozen density interactions are more natural than interactions between the QM and point-charges, and thus, this approach avoids the over-polarization. Boundary atoms represented by frozen electron density can be extended further from the QM region and can create smoother boundaries between the QM and MM regions. Ideally, a system could be constructed with a large boundary region between the QM and MM regions (see Figure 3.3).

3.6 Monte Carlo simulations

3.6.1 Stochastic sampling

Monte Carlo simulations sample phase space by generation random changes to the system. Completely random sampling could eventually find a global minimum, however, majority of the random structures would have energies well above kT and contribute very little to the statistical averages.

Almost any type of change can be made to the system in a Monte Carlo simulation. The primary limitations are the creativity of the programmer and the efficiency of making the changes. This allows Monte Carlo simulations to be tailored to the system and focus on the randomly changing the most important features of a system.

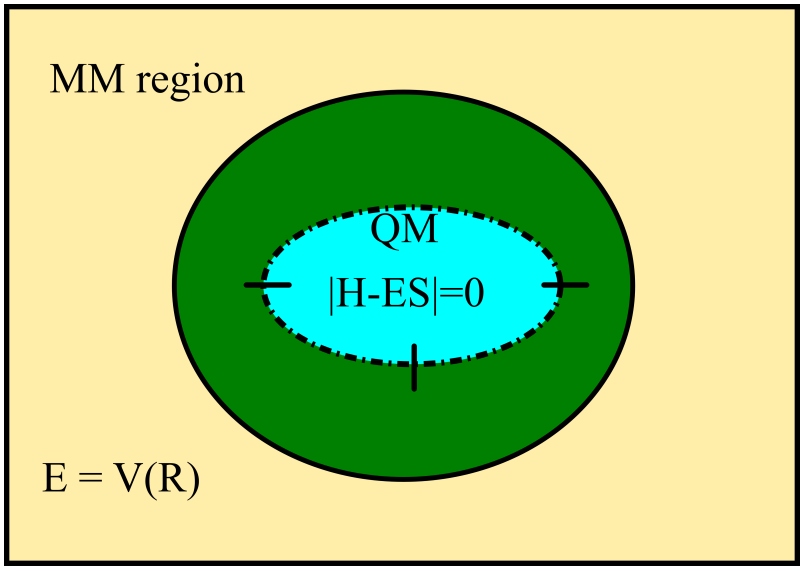


Figure 3.3: Schematic of a QM/MM simulation box with a large boundary region between the QM and MM regions.

If the moving atoms and/or type of random change are chosen randomly, this procedure satisfies "detailed balance." I.e. the move from configuration i to configuration j can be reversed by a move from configuration j to configuration i.

3.6.2 Canonical ensemble

A more efficient approach is to accept or reject the random moves based on the Boltzmann weight of the configuration.

$$P_{acc} \propto e^{-\Delta E \beta} \ , \tag{3.52}$$

where P_{acc} is the probability of accepting the move, ΔE is the change in energy, and β is the inverse temperature,

$$\beta = \frac{1}{kT} \ . \tag{3.53}$$

This proceedure leads to biased sampling of the lower energy states.

Unlike molecular dynamics simulations, the natural ensemble for Monte Carlo simulations is the canonical ensemble (NVT). This is advantageous since there is no need to couple the simulation to a thermostat.

3.6.3 Isobaric-isothermal ensemble

Although NVT simulations are natural for Monte Carlo, it is relatively easy to change to the NPT ensemble. NPT simulations are performed by randomly changing the volume of the simulation box. This procedure produces a slightly different expression for the probabilities,

$$P_{acc} \propto e^{-(P\Delta V + \Delta E)\beta + N\Delta ln(V)} \ , \tag{3.54}$$

where P is the pressure, ΔV is the change in volume, and N is the number of atoms.

3.7 Path-integral Monte Carlo

3.7.1 Path-integral formalism

Coming soon...

3.7.2 PIMC simulations

Path-integral Monte Carlo simulations are performed by simulating a large number of classical systems which are then coupled together via harmonic bonds. Allowed Monte Carlo moves include random displacements of all beads in a single atom, translations of an entire centroid, and volume changes.

The spring energy between the beads represents the quantum kinetic energy of the centroid. Moves are accepted based on the effective energy, E_{eff}

$$E_{eff} = E_{spring} + \frac{1}{N_p}\sum_{i}^{N_p} E_{pot,i} \ , \tag{3.55}$$

where the spring energy is given by

$$E_{spring} = \sum_{j}^{N_a} \frac{m_j N_p}{\hbar^2 \beta^2} \sum_{i}^{N_p} (r_i - r_{i-1})^2 \ . \tag{3.56}$$

Here the N_a is the total number of atoms, N_p is the number of beads, r_i is the position of bead i, and E_{pot} is the potential energy. The acceptance probability is given by

$$P_{acc} \propto e^{\Delta E_{eff}\beta} \ . \tag{3.57}$$

The path-integral total energy is slightly different from the effective potential.

$$E_{pi} = \frac{3N_a N_p}{2\beta} - E_{spring} + \frac{1}{N_p}\sum_{i}^{N_p} E_{pot,i} \ , \tag{3.58}$$

where E_{pi} is the total energy. The first two terms in Eq. 3.58 represent the classical kinetic energy for all particles and the amount of kinetic energy absorbed by centroid harmonic bonds.

3.7.3 Ergodicity

Classical Monte Carlo simulations typically scale as N_a^2 due to the pairwise force field interactions of the atoms. Since path-integral simulations have N_p simulations that run in tandem, the overall scaling is $N_p N_a^2$. In addition to making the energy calculations computationally more expensive, the efficiencies of the Monte Carlo moves tend to decrease. Monte Carlo moves typically consist of changes to the positions of a single atom or possibly more complicated moves where bond angles are shifted. This leads to large systems changing very slowly and the path-integral formalism essentially increases the size of the system.

Another problem introduced by the path-integral formalism comes from the harmonic constraints between the beads. The force constants for the path-integral springs is proportional to the mass of the particle, which means that heavy atoms can experience very strong forces holding the centroid close together. The presence of strong potentials restricts the movement of the atoms, and reduces P_{acc}.

Bibliography

[1] Michael J. Frisch, G. W. Trucks, H. Bernhard Schlegel, Gustavo E. Scuseria, Michael A. Robb, James R. Cheeseman, Giovanni Scalmani, Vincenzo Barone, Benedetta Mennucci, G. A. Petersson, H. Nakatsuji, M. Caricato, Xiaosong Li, H. P. Hratchian, Artur F. Izmaylov, Julien Bloino, G. Zheng, J. L. Sonnenberg, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, J. A. Montgomery Jr., J. E. Peralta, F. Ogliaro, Michael J. Bearpark, Jochen Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, Rika Kobayashi, J. Normand, Krishnan Raghavachari, Alistair P. Rendell, J. C. Burant, S. S. Iyengar, Jacopo Tomasi, M. Cossi, N. Rega, N. J. Millam, M. Klene, J. E. Knox, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, R. L. Martin, K. Morokuma, V. G. Zakrzewski, G. A. Voth, P. Salvador, J. J. Dannenberg, S. Dapprich, A. D. Daniels, Ö Farkas, J. B. Foresman, J. V. Ortiz, J. Cioslowski, and Douglas J. Fox. Gaussian 09 rev. d.01, 2009.

[2] Justin M. Turney, Andrew C. Simmonett, Robert M. Parrish, Edward G. Hohenstein, Francesco A. Evangelista, Justin T. Fermann, Benjamin J. Mintz, Lori A. Burns, Jeremiah J. Wilke, Micah L. Abrams, Nicholas J. Russ, Matthew L. Leininger, Curtis L. Janssen, Edward T. Seidl, Wesley D. Allen, Henry F. Schaefer, Rollin A. King, Edward F. Valeev, C. David Sherrill, and T. Daniel Crawford. Psi4: An open-source ab initio electronic structure program. *WIREs Comput. Mol. Sci.*, 2(4):556–565, 2012.

[3] M. Valiev, E.J. Bylaska, N. Govind, K. Kowalski, T.P. Straatsma, H.J.J. Van Dam, D. Wang, J. Nieplocha, E. Apra, T.L. Windus, and W.A. de Jong. Nwchem: A comprehensive and scalable open-source solution for large scale molecular simulations. *Computer Physics Communications*, 181(9):1477 – 1489, 2010.

[4] J. W. Ponder. Tinker, software tools for molecular design, version 7.0. Washington University: St. Louis, MO, 2015.

[5] Steve Plimpton. Fast parallel algorithms for short-range molecular dynamics. *J. Comp. Phys.*, 117(1):1–19, 1995.

[6] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 3 edition, 2007.

[7] Daniel Sheppard, Rye Terrell, and Graeme Henkelman. Optimization methods for finding minimum energy paths. *The Journal of Chemical Physics*, 128(13):134106, 2008.

[8] Graeme Henkelman, Blas P. Uberuaga, and Hannes Jnsson. A climbing image nudged elastic band method for finding saddle points and minimum energy paths. *The Journal of Chemical Physics*, 113(22):9901–9904, 2000.

[9] Graeme Henkelman and Hannes Jnsson. Improved tangent estimate in the nudged elastic band method for finding minimum energy paths and saddle points. *The Journal of Chemical Physics*, 113(22):9978–9985, 2000.

[10] Steven K. Burger and Weitao Yang. Quadratic string method for determining the minimum-energy path based on multiobjective optimization. *The Journal of Chemical Physics*, 124(5):054109, 2006.

[11] Li Xie, Haiyan Liu, and Weitao Yang. Adapting the nudged elastic band method for determining minimum-energy paths of chemical reactions in enzymes. *The Journal of Chemical Physics*, 120(17):8039–8052, 2004.

[12] G. Andrés Cisneros and Weitao Yang. *Comparison Of Reaction Barriers In Energy And Free Energy For Enzyme Catalysis*, pages 57–78. Springer Netherlands, Dordrecht, 2009.

[13] G. Andrés Cisneros, Mikko Karttunen, Pengyu Ren, and Celeste Sagui. Classical electrostatics for biomolecular simulations. *Chem. Rev.*, 114(1):779–814, 2014.

[14] Christian Kramer, Alexander Spinn, and Klaus R. Liedl. Charge anisotropy: Where atomic multipoles matter most. *J. Chem. Theory Comput.*, 10(10):4488–4496, 2014.

[15] William L. Jorgensen, Jayaraman Chandrasekhar, Jeffry D. Madura, Roger W. Impey, and Michael L. Klein. Comparison of simple potential functions for simulating liquid water. *J. Chem. Phys.*, 79(2):926–935, 1983.

[16] Michael W. Mahoney and William L. Jorgensen. A five-site model for liquid water and the reproduction of the density anomaly by rigid, nonpolarizable potential functions. *J. Chem. Phys.*, 112(20):8910–8922, 2000.

[17] Marie L. Laury, Lee-Ping Wang, Vijay S. Pande, Teresa Head-Gordon, and Jay W. Ponder. Revised parameters for the amoeba polarizable atomic multipole water model. *J. Phys. Chem. B*, 119(29):9423–9437, 2015.

[18] Yue Shi, Zhen Xia, Jiajing Zhang, Robert Best, Chuanjie Wu, Jay W. Ponder, and Pengyu Ren. Polarizable atomic multipole-based amoeba force field for proteins. *J. Chem. Theory Comput.*, 9(9):4046–4063, 2013.

[19] Anthony J. Stone. *The Theory of Intermolecular Forces*. Oxford University Press, 2013.