TINKTEP and polarisable embedding: how to do linear-scaling QM/polarisable-MM with ONETEP and TINKER.

Jacek Dziedzic

v1.0 May 2018, last updated May 2018

This manual pertains to ONETEP versions v4.5.18.8 and later, and TINKTEP versions v1.19 and later.

1 Executive summary

TINKTEP[3, 4] is an interface between ONETEP and TINKER (an implementation of the AMOEBA force-field). It enables QM/MM calculations, where ONETEP is used as the QM engine, and TINKER is used as the MM engine. The two main distinguishing features of TINKTEP are the linear scaling of the QM part (subject to usual ONETEP linear scaling caveats), and the fact that the MM side uses a polarisable force field (AMOEBA). There is full mutual self-consistency between the QM and MM subsystem – the QM subsystem polarises the MM subsystem and vice versa. Van der Waals interactions between QM and MM are also taken care of. TINKTEP is work in progress, both on the front of theory (improving the model) and implementation (adding desired features, simplifying use). It is currently not distributed to end users and not fully supported at the level we support mainstream ONETEP features.

2 Main limitations

Some of these might be deal-breakers for you, so make sure you read this carefully.

- You must obtain and install TINKER separately, and then patch it using the provided patches.
- The filesystem on the node from which you submit parallel jobs must be visible on the compute nodes. This can make set-up tricky on systems like ARCHER, where there is an intermediate layer of MOM nodes between the login node and the compute nodes. Currently it's easiest to run TINKTEP on single-node desktop machines, where you can leverage MPI and OMP parallelism without the hassle of a batch system.
- Only open boundary conditions (OBCs) are supported so far. The QM calculation is performed in OBCs, with the QM subsystem embedded in MM point charges, dipoles, quadrupoles and polarisable dipoles. The MM calculation is performed in OBCs. You will not be able to simulate e.g. a QM subsystem in a periodic box of water, for instance, but using a large sphere of MM water molecules is a decent workaround. We plan to support PBCs in the future.
- QM/MM forces, molecular dynamics, geometry optimisation and transition state search are all not available currently.
- Covalent bonds spanning the QM/MM interface are not supported.
- All QM species must have identical NGWF radii. Without this simplification the numerical methods used in SWRI (refer to "Spherical-wave resolution of identity, distributed multipole analysis (DMA) and Hartree-Fock exchange" manual for more details) would get ugly. Typically this is a non-issue, just increase the NGWF radii to the largest value. This might admittedly be an issue if you have a single species that requires a large NGWF radius. This is checked against and ONETEP will not let you do SWRI unless all NGWF radii are identical.
- TINKTEP is currently incompatible with complex NGWFs and ONETEP will refuse to use both.

- The TINKTEP1 model uses a classical model for QM/MM van der Waals interactions. The TINKTEP2 model uses a quantum-classical model for QM/MM van der Waals interactions, which requires suitable vdW parameters for all MM species that you use. We only provide these parameters for water, K^+ and Cl^-, so applications of TINKTEP2 are restricted to embedding a QM subsystem in water or KCl solutions, unless you wish to determine suitable MM vdW parameters yourself. These are **not** the familiar LJ parameters σ and ϵ .

3 Basics

Consult [3] for a detailed exposure of the theory behind the model. Consult [4] for a description of TINKTEP2 and how it differs from TINKTEP1. This manual takes a hands-on approach, but assumes you are familiar with the theory. This manual should be suitable for both TINKTEP1 and TINKTEP2, but the former will likely be discontinued in the future.

TINKTEP uses Distributed Multipole Analysis (DMA)[1, 2, 5] to build an auxiliary representation (termed QM*) of the QM subsystem as a set of multipoles. Make sure you are famliar with DMA and SWRI ("Spherical-wave resolution of identity, distributed multipole analysis (DMA) and Hartree-Fock exchange") beforehand.

4 Setting up your computational environment

1. Obtain a copy of TINKER v7.1.3. This should be available at no cost from Jay Ponder's website. It is imperative that you use v7.1.3 or else the patches supplied with ONETEP will not work.
2. Patch your copy of TINKER using the set of patches located in `utils/tinktep/tinktep_patches`. Make sure they all applied correctly.
3. Compile and install your patched copy of TINKER.
4. Ensure that at least the following executables from the TINKER suite can be found in your `PATH`: `analyze`, `dynamic`, `poledit`.
5. Compile ONETEP v4.5.18.8 or later.
6. Ensure all scripts supplied in `utils/tinktep` can be found in your `PATH`.

5 Setting up a QM/MM calculation

5.1 Input `.xyz` file

First, prepare your complete QM+MM system in the form of a TINKER `.xyz` file. This format differs from the four-column (species, x, y, z) `.xyz` file you might be familiar with. Consult the TINKER manual for the details of the format. In further text "`.xyz` format" will implicitly mean the TINKER `.xyz` format. If your `.xyz` file came from a TINKER MD simulation, no adjustments are necessary. If you are creating the `.xyz` file on your own, make sure you get the atom types and classes right, and the connectivity too – as prescribed by your MM `.prm` file (e.g. `amoeba09.prm`). If you do not know what atom types to choose for what will be your QM atoms, do not worry particularly, just make sure they roughly match their classical equivalents in the `.prm` file in terms of chemical species, hybridisation (sp2, sp3, etc.) and connectivity. What will only matter will be their polarisabilities (for correct Thole damping of induced QM/MM interactions) and classical vdW parameters (for QM/MM vdW energies (both the repulsive and dispersive term in TINKTEP1, dispersive term only in TINKTEP2).

An example `.xyz` file for a water dimer (one water molecule in QM, one water molecule in MM) could look like this:

```
    6
    1  O      0.583801    0.000000    0.759932    36     2     3
    2  H      0.000000    0.000000    0.000000    37     1
    3  H      0.000000    0.000000    1.530090    37     1
    4  O     -0.687305    0.000000    2.795684    36     5     6
    5  H     -0.448269   -0.763921    3.325671    37     4
    6  H     -0.448269    0.763921    3.325671    37     4
```

In the above 36 and 37 are atom types as defined in `amoeba09.prm`, and the last two colums define connectivity. Do not worry about the absolute positioning of the molecule (if it crosses zero in any of the directions) – TINKER will work in OBC mode and will not attempt to wrap your atoms back to the simulation cell, because there is none. ONETEP will work with a suitably translated version of the molecule anyway.

5.2 Input `.tag` file

Now we need to designate each atom as part of the QM subsystem or the MM subsystem. This is done via a `.tag` file. This file should contain two or three lines. The first line specifies the indices of atoms belonging to the QM subsystem. The second line specifies the indices of atoms belonging to the MM subsystem, like this:

```
1 2 3
4 5 6
```

The above assigns the first water molecule to the QM subsystem, and the second water molecule to the MM subsystem. In the `.tag` file, all atoms must be accounted for. If you want TINKTEP to ignore some atoms (say, you have `.xyz` file of a large system and want to discard some of it), put their indices in the third line. Normally you would simply omit the third line. It is not permitted for covalent bonds to span the QM/MM interface, and TINKTEP will refuse to proceed if it detects this. For instance this:

```
1 2
3 4 5 6
```

would not be a valid `.tag` file.

If you want *no* atoms in the QM subsystem (for a purely MM calculation) or the MM subsystem (for a purely QM calculation), put `-1` in the corresponding line, rather than leaving it blank.

Do not put any comments or additional information in the `.tag` file, that would make it misformatted.

Rename your `.tag` file to use the same base name as the `.xyz` file (say, `my_molecule.tag` and `my_molecule.xyz`).

5.3 Input `.key` file

Create a `.key` file with the same base name as the `.xyz` and `.tag` files, and with the following contents:

```
digits 9
parameters amoeba09
```

The line with `digits 9` is necessary to force TINKER to use extra precision in its outputs. The second line specifies the MM parameter file for TINKER. Adjust it if you want to use a file different from `amoeba09.prm`.

5.4 Input .dat.template file

Create a .dat.template file with the same base name as the .xyz, .tag and .key files. Populate this file with the keywords you want to be passed to ONETEP. Essentially, this file will be slightly modified by TINKTEP (specifically by qm_xyz_to_dat) and will become the .dat that ONETEP will read. The modifications undertaken by TINKTEP are:

- pol_emb_pot_filename will be added[1] to instruct ONETEP to perform a QM/MM calculation and inform it about the name of the file used for communicating between ONETEP and TINKER.
- pol_emb_polscal will be set accordingly if qm_mm_polscal was set in tinktep.config.
- pol_emb_thole_a will be set accordingly if qm_mm_thole_a was set in tinktep.config.
- pol_emb_fixed_charge T will be added if the MM force field is *not* polarisable (e.g. GAFF), to inform ONETEP about this fact.
- A %block positions_abs will be added, containing the species and positions of QM atoms inferred from the .xyz and .tag files, suitably translated.
- If tinktep.config specified qm_thole_polarisability, a %block thole_polarisabilities will be added, containing the Thole polarisabilities of QM atoms inferred from the .prm and .tag files.
- If the scenario of a purely QM calculation with classical atoms ("sparkles") has been selected by specifying classical_atoms in tinktep.config, a %block classical_info will be added, containing the species and positions of MM atoms inferred from the .xyz and .tag files, suitably translated.
- If the .xyz file contained a bounding box (for PBC calculations), a suitable %block lattice_cart will be added to match the MM box size. PBCs are not supported yet, do not rely on this functionality.

Basically, what you put in the .dat.template file should look like a normal ONETEP .dat file, *except for* the positions of atoms. Do not attempt to create a .dat file on your own, leave it to TINKTEP to create it automatically when it is run. For instance, for our water dimer example you could use this bare-bones .dat.template file:

```
! --- usual ONETEP keywords ---
%block species_atomic_set
H  "SOLVE"
O  "SOLVE"
%endblock species_atomic_set

%block species
H  H 1 1 7.0
O  O 8 4 7.0
%endblock species

%block species_pot
H  'H_04.recpot'
O  'O_02.recpot'
%endblock species_pot

%block lattice_cart
30.0  0.0  0.0
 0.0 30.0  0.0
 0.0  0.0 30.0
%endblock lattice_cart

cutoff_energy 1000 eV
charge 0
xc_functional PBE
```

[1]Except for purely QM calculations.

```
 dispersion 1

 ! --- cutoff Coulomb to enforce OBCs in ONETEP ---
 coulomb_cutoff_type SPHERE
 coulomb_cutoff_radius 40.0 bohr
 coulomb_cutoff_write_int F

 ! --- DMA setup, needed for QM/MM. Consult DMA manual for details ---
 %block swri
   for_dma 1 12 V 12 12 W
 %endblock swri

 %block species_swri-for_dma
 H
 O
 %endblock species_swri-for_dma

 dma_calculate T
 dma_use_ri for_dma
 dma_max_l 1
 dma_max_q 12
 dma_metric ELECTROSTATIC
 dma_bessel_averaging T
 dma_scale_charge F

 ! --- Polarisable embedding, needed for QM/MM. See further text. ---
 pol_emb_dma_min_l 0
 pol_emb_dma_max_l 1
 pol_emb_mpole_exclusion_radius 1.00 bohr
 pol_emb_repulsive_mm_pot_cutoff 10.0 bohr

 %block mm_rep_params
 H   35 2.400 ! follows TINKTEP-2 paper
 O  550 1.580 ! follows TINKTEP-2 paper
 %endblock mm_rep_params
```

5.5 tinker's .prm file

Copy the .prm file of your choice (typically amoeba09.prm) to the same directory where you put the .xyz, .tag, .key and .dat.template files. Do not rename it. Ensure its basename is reflected in the parameters keyword in the .key file.

5.6 tinktep.config file

This is the main file for controlling the QM/MM calculation. Its name is fixed, do not change it. Here's an example suitable for our water dimer, using the TINKTEP2 model:

```
jobname water_dimer

# *** Computational environment set-up ***
tinker_nthreads 8
onetep_nranks 2
onetep_nthreads 8
onetep_executable ./onetep.RH7
mpirun_executable mpirun
```

```
# *** Nuts and bolts of the QM/MM interface ***
qm_mm_polscal 6.0
qm_polarisability
qm_thole_polarisability
renumber_offset 500

# *** Physics ***

# Undamped fixed, permanent multipoles using the full density representation,
# and Thole-damped induced dipoles using the QM* representation. MM repulsive potential.
onetep_perm_mpoles     perm_fix_rep potential_coulombic_smeared energy_from_potential
onetep_induced_dipoles ind_qmstar   potential_thole_damped      energy_from_potential

# TINKER handles all bonded (valence) terms between MM atoms.
tinker_bond_energy 1
tinker_angle_energy 1
tinker_ureybrad_energy 1

# TINKER handles MM electrostatics.
tinker_mm_perm_energy 1
tinker_mm_pol_energy 1

# ONETEP handles QM/MM electrostatics.
tinker_qm_mm_perm_energy 0
tinker_qm_mm_pol_energy 0

# TINKER handles van der Waals for MM, and only the dispersive term for QM/MM.
tinker_mm_vdw_energy 1
tinker_qm_mm_vdw_energy 2
```

All `tinktep.config` keywords will be described later.

6 Running a QM/MM calculation

Once you have all input files in place, simply type `tinktep` to run the QM/MM calculation. Expect the following to happen:

1. `xyz_split` will be run to split your `.xyz` file into a `qm.xyz` and a `mm.xyz` file, based on the contents of the `.tag` file.
2. `qm_xyz_to_dat` will be run to build a ONETEP `.dat` file from the `.dat.template` file and the `qm.xyz` file, using information from the `.prm` file.
3. A pair of FIFOs (`$QM2MM.lock` and `$MM2QM.lock`) will be created. These will be used for interprocess communication (ONETEP to TINKTEP and TINKTEP to ONETEP).
4. ONETEP will be started in the background (via `mpirun` or equivalent).
5. A watchdog process will be started in the background. It will keep an eye on the `mpirun` process that launched ONETEP and on ONETEP's `.err` and `.error_message` files. It will attempt to clean up gracefully if it decides that ONETEP crashed or was killed.
6. TINKTEP will block until it ONETEP reaches a point where total energy needs to be evaluated. Then it will resume.
7. TINKTEP will read the `.gdma_like.txt` file produced by ONETEP. This file contains the multipole representation of the QM subsystem. It will run TINKER's `poledit` to process these multipoles.
8. `xyz_process` will be run to process the `qm.xyz` and `mm.xyz` files to a form digestible by TINKER (`qm_mm.xyz` file). This is mostly about renumbering the atom types in the QM subsystem so that

they do not clash with the types in the `.prm` file.

9. `key_process` will be run to prepare a suitable `.key` file for TINKER (`qm_mm.key`). This takes the contents of the original `.key` file, and modifies it accordingly so that it is digestible by TINKER. For instance dummy bond and angle parameters will be supplied for the QM atoms, QM sites and multipoles will be renumbered, polarisabilities of QM atoms will be defined, the QM subsystem will be made inactive and "only formally polarisable", etc..
10. TINKER (specifically `analyze` and `dynamic`) will be run to obtain the polarisation response of the MM subsystem, all of MM electrostatics, QM/MM electrostatics, QM/MM van der Waals energies, MM van der Waals and MM bonded interactions. Not all of these terms will necessarily be used in the final energy expression.
11. `mpoles_process` will be run to process TINKER's multipoles and energy terms into a format understandable by ONETEP (`.mpoles_for_onetep` file).
12. ONETEP will resume, after having been pinged via `$MM2QM.lock`.
13. If SCF convergence has been reached, TINKTEP will terminate with a short summary. If not, control will transfer to step 6.

All in all, the TINKTEP script drives both ONETEP and TINKER. ONETEP is executed once, in the background, and is resumed when necessary. TINKER is started each time ONETEP performs an energy evaluation. TINKER, which does not support MPI parallelism, is run on the local node (possibly using OMP threads). ONETEP is started via `mpirun` or equivalent, and it's the user's responsibility to set the parallel environment in such a way, that ONETEP gets started on the appropriate nodes (e.g. via a hostfile).

All ONETEP output goes to a `qm_mm.out` file with the same base name as the input. All ONETEP error messages go to a `qm_mm.err` file with the same base name as the input. TINKTEP's output goes to `stdout` and `stderr`. TINKTEP attempts to detect a large variety of error conditions and should at least provide an informative error message if something goes wrong. When diagnosing errors, examine `stderr`, ONETEP's `qm_mm.err` file, ONETEP's `qm_mm.error_message` file (if any), and see if there's a file called `error_message` (with no base name) – it is created when more exotic error conditions occur.

All intermediate files are automatically moved to a subdirectory called `intermediate` (at each SCF iteration), they are tagged with an SCF iteration (energy evaluation) number. It is safe to delete this directory after the calculation has run, it's mostly useful when diagnosing problems.

7 Output from a QM/MM calculation

Your qm_mm.out file will contain usual ONETEP output, interspersed with a lot of informative messages from DMA and polarisable embedding. Every time the energy is evaluated, you will get a detailed breakdown of energies associated with the QM/MM interface. Here's what it looks like:

```
/~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~\
| Polarisable embedding potential from water_dimer_mm.mpoles_for_onetep       |
| Multipole set "perm_fix_rep": 6 sites.                                      |
| Multipole set "ind_qmstar": 6 sites.                                        |
| All in all 12 sites, and 11 external energy terms (out of which 3 #ignored). |
| Energy term                           Energy      Source   Included?       |
| - MMv bond stretch                  0.000013814 Ha  TINKER      YES         |
| - MMv angle bend                    0.000229041 Ha  TINKER      YES         |
| - MMv Urey-Bradley                 -0.000000274 Ha  TINKER      YES         |
| - #QM/MM perm mpole                -0.032818501 Ha  TINKER       NO         |
| - #QM/MM polarisation              -0.000794393 Ha  TINKER       NO         |
| - MM perm mpole                     0.000000000 Ha  TINKER      YES         |
| - MM+ polarisation                  0.000000000 Ha  TINKER      YES         |
| - #QM/MM vdW-rep                    0.050008920 Ha  TINKER       NO         |
| - QM/MM vdW-disp                   -0.013332666 Ha  TINKER      YES         |
| - MM vdW-rep                        0.000000000 Ha  TINKER      YES         |
| - MM vdW-disp                       0.000000000 Ha  TINKER      YES         |
| - QM elec <-> rep MM perm_fix       0.033334141 Ha  ONETEP   REPULS  P Fr   |
| - QM elec <-> MM perm_fix_rep       0.229940160 Ha  ONETEP   COUL-S  P Fr   |
| - QM core <-> MM perm_fix_rep      -0.263586238 Ha  ONETEP   COUL-S  P FR   |
| - QM* elec <-> MM ind_qmstar        0.014117659 Ha  ONETEP    THOLE  I*     |
| - QM core <-> MM ind_qmstar        -0.014912052 Ha  ONETEP    THOLE  I*     |
|------------------------------------------------------------------------------|
|                     External |          Internal |         Difference      |
| Permanent:       -0.032818501 |      -0.033646079 |     0.000827577479 (Ha) |
| Induced:         -0.000794393 |      -0.000794393 |    -0.000000000001 (Ha) |
|------------------------------------------------------------------------------|
| Perm embed. potential  min: -0.3743E+01  max:  0.6338E+01  norm:  0.1776E-01 |
\~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~/
```

From the above example you can infer the following:

1. The file from which ONETEP reads the details of the MM embedding is water_dimer_mm.mpoles_for_onetep.
2. There are two sets of multipoles, with 6 sites each. The first set entails permanent (perm) (as in not induced), fixed (fix) (as in not variable in time) multipoles that generate a repulsive (rep) potential. The second set entails induced (ind) multipoles, which interact not with ONETEP's full electronic density, but with the QM* description (qmstar). You probably expected 3, not 6 sites, but the .mpoles_for_onetep file also includes QM sites in addition to MM sites. The QM sites are tagged with "#" and subsequently ignored.
3. There are 11 energy terms coming from TINKER, but 3 of these will be ignored by ONETEP in accordance with the user's wishes. The ignored terms are prefixed by "#" and have a "NO" in the "Included?" column. The ignored terms in this case are:
   - QM interaction with permanent MM multipoles as calculated by TINKER. This is because we instead use the full density QM representation (as calculated by ONETEP) for this term, excluding TINKER's approximate idea on purpose (compare tinker_qm_mm_perm_energy 0 earlier).
   - QM interaction with induced MM multipoles as calculated by TINKER. This is because we instead use the value calculated by ONETEP for this term, even though the two should be (and are) identical, excluding TINKER's value on purpose (compare tinker_qm_mm_pol_energy 0 earlier).
   - Repulsive part of QM/MM van der Waals interaction as calculated by TINKER. This is because we instead use the repulsive potential model introduced in TINKTEP2, calculated by ONETEP for

this term, excluding TINKER's classical value on purpose (compare `tinker_qm_mm_vdw_energy 2` earlier).

The other terms coming from TINKER are included. These are: MM valence terms (`MMv`): bond, angle and Urey-Bradley, MM-MM permanent multipole interactions (zero in this case, as there is only one molecule in MM, and AMOEBA masked the MM permanent interactions within the water molecule), MM polarisation ("+" serves as a reminder that this polarisation is not strictly only due to MM, because of non-additivity) (again zero in this case, because of masking), dispersive part of QM/MM van der Waals interactions, and MM/MM vdW terms (repulsion and dispersion) (also zero, since there is only one molecule in MM).

4. There are 5 energy terms coming from ONETEP (denoted by "ONETEP" in the "Source" column). These are
   - The interaction of QM electrons with the MM repulsive potential attached to one of the multipole sets (`REPULS`).
   - The interaction of QM electrons with the permanent MM multipoles, treated Coulombically with smearing (`COUL-S`).
   - The interaction of QM ionic cores with the permanent MM multipoles, treated Coulombically with smearing (`COUL-S`).
   - The interaction of QM* electrons (i.e. electronic multipoles) with the induced MM multipoles, treated using Thole damping (`THOLE`).
   - The interaction of QM ionic cores (i.e. ionic charges) with the induced MM multipoles, treated using Thole damping (`THOLE`).
5. The symbols to the right of the table inform us about the assumptions ONETEP makes about some energy terms:
   - Column 1: `P` – MM multipole set is permanent, or `I` – MM multipole set is induced. This affects energy expressions – induced multipoles require work to assemble, cf. Ref.[3].
   - Column 2: `*` – calculation uses the QM* representation of the density, or (blank) – calculation uses the full density.
   - Column 3: `F` – MM multipole set is fixed (its value is time-independent), so its electrostatic potential can be stored and reused, or (blank) – MM multipole set is not fixed (then its electrostatic potential has to be recalculated every time).
   - Column 4: `1` – Energy term has been calculated for the first time, and will either be reused later (e.g. for QM cores interacting with permanent MM multipoles) or at least the MM potential will be reused (e.g. for QM electrons interacting with permanent MM multipoles), or `R` – energy term has just been reused, or `r` – the MM potential has been reused, but the energy has been recalculated, or (blank) – neither of the above.
   - Column 5: `S` – `dma_multipole_scaling` affected this energy term, or `s` – `pol_emb_perm_scaling` affected this energy term. These are expert directives, do not worry about them.
6. This is followed by a summary of QM/MM permanent and QM/MM induced electrostatics. "External" is TINKER's idea of these energy terms, "Internal" is ONETEP's idea, "Difference" is the difference between the two. Unless you do something very exotic, like ignoring polarisation, the row with "Induced" should always match extremely well, because both ONETEP and TINKER use the same model (QM* interacting with MM dipoles), and their calculations should match (if not, this indicates a bug or a set-up problem, and ONETEP will abort). Unless you do something exotic, like using the QM* for permanent interactions, the row with "Permanent" will not match, because TINKER uses the QM* model and thus suffers from charge penetration, while ONETEP uses the full density for permanent interactions, arriving at the "right" result. Here, "Difference" is a good estimate of QM/MM charge penetration error.
7. The last row gives some statistics about the permanent MM multipole potential in which QM electrons are embedded.

8 `tinktep.config` directives

Here is a list of directives understood by TINKTEP that you can put in the `tinktep.config` file. Make sure you spell these right, unlike ONETEP, TINKTEP **silently ignores** directives it does not recognise. You can use "#" to denote comment lines. These will be ignored.

8.1 Environment set-up

`jobname (string)` [**mandatory, basic**] – specifies the base name for input (`.xyz`, `.tag`, `.key`, `.dat.template`), files. Example: `jobname qm_tryptophan_in_40_mm_waters`.

`onetep_executable (string)` [**mandatory, basic**] – specifies the name of the ONETEP executable that TINKTEP will pass to `mpirun` (or equivalent). This file must be user-executable.

`onetep_nranks (integer)` [**mandatory, basic**] – specifies the number of MPI ranks that TINKTEP will tell `mpirun` (or equivalent) to start ONETEP on.

`onetep_nthreads (integer)` [**mandatory, basic**] – specifies the number of OMP threads that TINKTEP will tell ONETEP to use (by setting `OMP_NUM_THREADS`. You can always override this with specific ONETEP thread keywords in the `.dat.template` file.

`onetep_args (string)` [**optional, expert**] – specifies additional arguments that you might want to pass to ONETEP. These will go between the ONETEP executable and the ONETEP input file. This only makes sense if your `onetep_executable` actually points to a wrapper script that will know what to do with these arguments.

`tinker_nthreads (integer)` [**optional, intermediate**] – specifies the number of OMP threads that TINKTEP will tell TINKER to use (by adding a keyword to `qm_mm.key`. If left unspecified, this will be left at TINKER's discretion. Caveat: TINKER sometimes carelessly outputs to `stdout` from OMP regions, which can cause a mess. If TINKTEP complains that it cannot parse TINKER's output, try setting this to 1 to disable OMP in TINKER.

`mpirun_executable (string)` [**mandatory, basic**] – specifies the name of the `mpirun` executable that TINKTEP will use to launch ONETEP. Set this to `mpirun`, unless your environment uses something fancier like `aprun`, or you want to specify a full path to select a specific `mpirun` executable.

`mpirun_args (string)` [**optional, intermediate**] – specifies additional arguments that you might want to pass to `mpirun`. These will go between the `mpirun` executable and `-np <onetep_nranks>`. Can be useful for passing a hostfile name.

`watchdog_unfazed_by_stderr (no args)` [**optional, intermediate**] – tells TINKTEP's watchdog not to keep an eye on ONETEP's `.err` file. Normally any output to this file is an indication that something went wrong, and the watchdog then initiates cleanup. In some environments you can get innocuous messages written to a job's `.err` file, e.g. warnings from MPI or the transport layer. Use this directive to immunize the watchdog against these.

8.2 QM/MM set-up: basic

`onetep_perm_mpoles (set_name) (potential_mode) (energy_mode)` [**mandatory, basic**] – specifies the treatment of permanent MM multipoles inside ONETEP. All three arguments are strings and are mandatory. These are extremely important and have to be discussed in detail.

- `set_name` – a short, descriptive name that will identify the permanent MM multipole set in ONETEP. Crucially, this name will also be parsed by ONETEP to infer the properties of this set. Thus, certain tokens (substrings) carry very specific meaning in the context of the name. These are:
  - `perm` – the set is permanent (as in "not induced"). This affects energy expressions – induced multipoles require work to assemble, permanent sets do not, cf. Ref.[3].
  - `ind` – the set is induced (as in "not permanent"). This affects energy expressions – induced multipoles require work to assemble, permanent sets do not, cf. Ref.[3].
  - `fix` – the set is fixed (as in "not changing in time"). This does not mean "not moving through space" (currently *all* MM multipoles are presumed not to be moving through space). Calculations on fixed sets will be optimised to re-use electrostatic potentials or entire energy terms.
  - `qmstar` – the set uses the QM* representation and not the full QM density when interacting with the QM subsystem. If absent, the full QM density is used by default.
  - `rep` – the set generates an MM repulsive potential (for the TINKTEP2 model). If absent, no MM repulsive potential will be generated by the set. Note: If `rep` is present for more than one set, only the last set set will generate the MM repulsive potential.
- `potential_mode` – describes how ONETEP generates the electrostatic potential coming from this multipole set. Four options are possible:
  - `potential_zero` – the set does not generate any potential (and so is essentially ignored).
  - `potential_coulombic_smeared` – the set generates a Coulombic potential, with a small degree of smearing only very close to the location of each point multipole – this is done to avoid singularities (cf. `pol_emb_mpole_exclusion_radius`, `polemb_smearing_a`).
  - `potential_coulombic_masked` – the set generates a Coulombic potential, which is, however, masked (zeroed) very close to the location of each point multipole – this is done to avoid singularities (cf. `pol_emb_mpole_exclusion_radius`). This is not recommended, except for tests, use `potential_coulombic_smeared` instead.
  - `potential_thole_damped` – the set generates a Thole-damped potential, mimicking AMOEBA polarisation interactions. Thole damping is a classical scheme and is designed to be applied to interactions between two point multipoles. Thus it is best suited to the QM* representation (cf. `qmstar` above), and not to interactions between MM point multipoles and the full, distributed QM density, although this is, in principle, possible. The magnitude of the damping depends on the polarisabilities of the two interacting sites. For MM sites the polarisabilities are determined by the force field (`.prm` file). For QM* sites the polarisabilities either have to be specified in the `.dat.template` file (`%block thole_polarisabilities`), or can be inferred automatically from the `.prm` file. The latter option is preferred, it can be activated via the directive `qm_thole_polarisability` in `tinktep.config`. This instructs TINKTEP (and `qm_xyz_to_dat` in particular) to add a suitable `%block thole_polarisabilities` automatically. When an attempt is made to use Thole damping with a multipole set that does not specify `qmstar`, a Thole-damped potential coming from the set will need to be integrated with the full QM density, and there is no corresponding polarisability that can be used in the Thole damping expression. In this unlikely scenario, the value of `pol_emb_pairwise_polarisability`, with a unit of bohr, is used for the Thole variable A (which is otherwise equal to $\sqrt{\alpha_1 \alpha_2}$). The default value of this parameter corresponds to the average polarisability of all atom types in AMOEBA09.
- `energy_mode` – describes how ONETEP calculates the electrostatic energy of this multipole set interacting with the QM subsystem. Two options are possible:
  - `energy_zero` – the set does not contribute to energy (and so is essentially ignored).

  - energy_from_potential – the set's contribution to energy will be made consistent with the setting for the potential (see above).

  Not all combinations of potential_mode and energy_mode make sense. For instance trying to combine potential_coulombic_smeared with energy_zero will lead to an inconsistency between the Hamiltonian and the energy expression, breaking LNV and NGWF convergence. Using energy_from_potential is generally the safest option, as it guarantees consistency.

onetep_induced_dipoles (set_name) (potential_mode) (energy_mode) [**mandatory, basic**] – specifies the treatment of induced MM dipoles inside ONETEP. The meaning of the arguments is the same as for onetep_perm_mpoles above.

Some typical settings for the above two keywords:

```
# Usual TINKTEP2 set-up: Permanent multipoles interact Coulombically with full QM density,
#                         induced dipoles interact with QM* via Thole damping,
#                         repulsive MM potential in effect (attached to perm. multipoles)
onetep_perm_mpoles      perm_fix_rep  potential_coulombic_smeared  energy_from_potential
onetep_induced_dipoles  ind_qmstar    potential_thole_damped       energy_from_potential

# Usual TINKTEP1 set-up: Permanent multipoles interact Coulombically with full QM density,
#                         induced dipoles interact with QM* via Thole damping,
#                         no repulsive MM potential in effect.
onetep_perm_mpoles      perm_fix      potential_coulombic_smeared  energy_from_potential
onetep_induced_dipoles  ind_qmstar    potential_thole_damped       energy_from_potential

# TINKTEP2 set-up for a non-polarisable force-field (eg. GAFF).
onetep_perm_mpoles      perm_fix_rep  potential_coulombic_smeared  energy_from_potential
onetep_induced_dipoles  ind_qmstar    potential_zero               energy_from_potential

# TINKTEP2 set-up, where both permanent and induced interactions use QM* and Thole damping
onetep_perm_mpoles      perm_fix_rep_qmstar  potential_thole_damped  energy_from_potential
onetep_induced_dipoles  ind_qmstar           potential_thole_damped  energy_from_potential
```

qm_thole_polarisability (no args) [**optional, basic**] – asks TINKTEP to automatically infer the Thole polarisabilities of QM sites from the .prm file and to automatically add a suitable %block thole_polarisabilities to the .dat file. Strongly recommended.

qm_polarisability (no args) [**optional, basic**] – forces TINKER to treat QM sites as "formally polarisable", that is, to take their polarisabilities into account in Thole damping expressions, but not to put induced dipoles on them. Treat this directive as mandatory, the alternative scheme is no longer supported.

8.3 QM/MM set-up: treatment of energy terms

The following directives control how different energy terms generated by TINKER are taken into account in the QM/MM calculation.

tinker_bond_energy (0 or 1) [**optional, intermediate**] – when enabled (1), the valence MM term due to bond stretches is included in the QM/MM energy expression. When omitted, defaults to 0. In typical scenarios you'd want this enabled. Disabling might be necessary if there are no bonds between MM atoms (e.g. for a noble gas).

tinker_angle_energy (0 or 1) [**optional, intermediate**] – when enabled (1), the valence MM term due to angle bends is included in the QM/MM energy expression. When omitted, defaults to 0. In typical scenarios you'd want this enabled. Disabling might be necessary if there are no angles between MM atoms (e.g. for a noble or diatomic gas).

tinker_ureybrad_energy (0 or 1) [**optional, intermediate**] – when enabled (1), the valence MM term due to Urey-Bradley interactions is included in the QM/MM energy expression. When omitted, defaults to 0. In typical scenarios you'd want this enabled. Disabling might be necessary if there are no Urey-Bradley interactions between MM atoms.

`tinker_mm_perm_energy (0 or 1)` [**optional, intermediate**] – when enabled (`1`), the electrostatic term due to permanent MM-MM interactions is included in the QM/MM energy expression. When omitted, defaults to `0`. In typical scenarios you'd want this enabled.

`tinker_mm_pol_energy (0 or 1)` [**optional, intermediate**] – when enabled (`1`), the electrostatic term due to MM-MM polarisation interactions is included in the QM/MM energy expression. Even though polarisation interactions are not additive, TINKER formally splits them *a posteriori* into QM-MM polarisation and MM-MM polarisation, which add up to total MM polarisation. When omitted, defaults to `0`. In typical scenarios you'd want this enabled.

`tinker_qm_mm_perm_energy (0 or 1)` [**optional, intermediate**] – when enabled (`1`), the electrostatic term due to interactions between permanent MM multipoles and QM is included in the QM/MM energy expression. When omitted, defaults to `0`. In typical scenarios you'd want this **disabled**, because TINKER's idea of this interaction suffers from charge-penetration error. This term would then be taken into account on ONETEP's side, via an `energy_from_potential` setting for permanent multipoles (cf. `onetep_perm_mpoles`).

`tinker_qm_mm_pol_energy (0 or 1)` [**optional, intermediate**] – when enabled (`1`), the electrostatic term due to QM-MM polarisation interactions is included in the QM/MM energy expression. Even though polarisation interactions are not additive, TINKER formally splits them *a posteriori* into QM-MM polarisation and MM-MM polarisation, which add up to total MM polarisation. When omitted, defaults to `0`. In typical scenarios you'd want this **disabled**. This term would then be taken into account on ONETEP's side, via an `energy_from_potential` setting for induced dipoles (cf. `onetep_induced_dipoles`). The two expressions should yield the same result, provided `qmstar` is used for `onetep_induced_dipoles`. Having ONETEP calculate this term permits checking the two results (TINKER's and ONETEP's) for consistency.

`tinker_mm_vdw_energy (0 or 1)` [**optional, intermediate**] – when enabled (`1`), the van der Waals term due to MM-MM non-bonded interactions is included in the QM/MM energy expression. When omitted, defaults to `0`. In typical scenarios you'd want this enabled.

`tinker_mm_vdw_energy (0 or 1 or 2 or 3)` [**mandatory, intermediate**] – controls if and how the van der Waals term due to QM-MM non-bonded interactions is included in the QM/MM energy expression. The following values are possible:

- 0 – omit QM-MM vdW interactions entirely.
- 1 – include QM-MM vdW interactions, both the repulsive and dispersive term, via classical Halgren potential. Recommended for TINKTEP1 model.
- 2 – include QM-MM vdW interactions, but only the dispersive term, via classical Halgren potential. Recommended for TINKTEP2 model, where repulsive QM-MM vdW interactions would be handled via a repulsive MM potential.
- 3 – include QM-MM vdW interactions, but only the repulsive term, via classical Halgren potential. Only included for completeness.

8.4 QM/MM set-up: details of the interface

The following directives control how the QM/MM interface behaves.

`qm_mm_polscal (real)` [**optional, intermediate**] – controls scaling of QM/MM polarisation interactions (introduced in TINKTEP2). Defaults to no scaling if omitted. The apparent polarisability of QM sites is scaled (multiplied) by the value of this parameter, thereby increasing damping if it is greater than 1.0, and attenuating damping (increasing the strength of polarisation interactions) if it is between 0.0 and 1.0. Recommended value: 6.0 for TINKTEP2, omit for TINKTEP1.

`renumber_offset (integer)` [**mandatory, intermediate**] – specifies the value by which atom types for QM atoms are offset from their original force field counterparts. This is used to avoid clashes between "fake" QM types presented to TINKER and original atom types. Use a large, three-digit value, like 500, unless you have good reason for doing otherwise.

`coord_xlat (x) (y) (z)` [**optional, intermediate**] – specifies the vector (in bohr) by which all QM atoms are translated in the `.dat` file. All arguments are real numbers and are mandatory. Values will be interpreted as bohr, do not specify any units. This directive becomes useful when the contents of the *qm.xyz* file are positioned unsuitably for ONETEP in OBC mode, e.g. leading to NGWFs not fitting in the simulation cell. In general, TINKER always works with original (untranslated) coordinates, and ONETEP always sees coordinates translated by this vector. All `.xyz` files work with the original coordinates, the `.dat` file works with the translated coordinates. If omitted, this value will be determined automatically: a bounding box will be calculated for the QM subsystem, and the translation vector will be such that the centre of the bounding box will coincide with the centre of the simulation cell. This is handy, but can lead to eggbox errors when comparing energies between different molecules (as they might be translated differently). It is advised to set this translation vector manually, and identically for all molecules whose energies will be compared.

8.5 QM/MM set-up: expert options

`classical_atoms (no args)` **[optional, expert]** – when present, TINKTEP will add a `%block classical_info` containing the species and positions of MM atoms inferred from the `.xyz` and `.tag` files, suitably translated, to the `.dat` file. Charges for the classical atoms will be extracted from specially crafted comments in the `.dat.template` file. This is useful when generating reference classical embedding ("sparkles") calculations. You should normally omit this directive. The format for specifying charges of classical atoms is as follows:

```
!$ classical_species_charge H 0.417
!$ classical_species_charge O -0.834
```

These lines are formally comments and will be ignored by ONETEP, but `qm_xyz_to_dat` will know how to interpret them.

`mm_fixed_charge (no args)` **[optional, expert]** – when present, instructs TINKTEP that the MM force field is not polarisable. Leave this directive out by default. Non-polarisable force-fields need a few particular tweaks (TINKER's `multipole` keyword is replaced by `charge` in the `qm_mm.key` file, the `polarizable` keyword needs to be omitted from same, `pol_emb_fixed_charge T` will be automatically added to the `.dat` file to instruct ONETEP not to look for `%block thole_polarisabilities` in the `.dat` file, and TINKER's output will be parsed differently). These tweaks are activated by this directive.

`pure_mm (no args)` **[optional, expert]** – when present, instructs TINKTEP that the calculation is a purely MM calculation, and ONETEP does not need to be invoked.

`pure_qm (no args)` **[optional, expert]** – when present, instructs TINKTEP that the calculation is a purely QM calculation: TINKTEP does not need to be invoked, and `pol_emb_pot_filename T` must **not** be added to the `.dat` file so that ONETEP runs without polarisable embedding. This is necessary due to a bug in TINKER, which causes it to hang if all atoms are inactive.

`qm_mm_thole_a (real)` **[optional, expert]** – when present, the specified value overrides the value of Thole's *a* parameter read from the `.prm` file. This value will be used *only* for QM/MM interactions, as TINKER will use the force-field value for MM/MM interactions.

`qm_dummy_atoms (no args)` **[optional, expert]** – experimental functionality for adding dummy DMA sites on MM atoms, do not use.

8.6 Command-line options to the tinktep script

`tinktep` is normally run without arguments. The following command-line options are supported:

`-1` – Only the first two stages described in Section 6 are performed, then TINKTEP exits. In essence, all input files are parsed and intermediate inputs are prepared, but neither ONETEP nor TINKER are actually run.

`-2` – All stages described in Section 6, *except* the first two, are performed. In essence, the calculation is run, assuming all intermediate inputs have been prepared in advance. Splitting the two parts of the calculation (preparation and actual execution) enables manual or scripted tweaking of intermediate inputs by the user.

`--dry-run` – equivalent to `-1`, only deprecated.

9 ONETEP keywords pertaining to polarisable embedding

A number of ONETEP keywords can be used to control the polarisable embedding functionality, which underlies QM/MM calculations.

9.1 Keywords mandatory for polarisable embedding

`pol_emb_pot_filename (string)` – specifies the name of the file used for exchanging information between ONETEP and TINKTEP. This file will be read by ONETEP and constructed by TINKTEP at every energy evaluation. This keyword is also used to turn on polarisable embedding (when it is present), and to turn it off (when it is absent). Default: absent.

`pol_emb_dma_max_l (integer)` – specifies the maximum angular momentum in the SW basis used in polarisable-embedding-DMA. In most scenarios this will be equal to l_{max} that you specified in the SWRI block. Read the description of l_{max} in the DMA documentation to understand the meaning of this parameter. You can use a lower value than the one specified in the SWRI block if you want to use only a subset of the SW basis set (e.g. for benchmarking). This keyword does not affect properties-DMA (for which `dma_max_l` should be used). This directly affects the quality of the QM* representation – specifying 0 leads to a charge-only representation, specifying 1 leads to a charge-and-dipole representation, specifying 2 leads to a charge-dipole-and-quadrupole representation.

9.2 Common optional keywords

`pol_emb_polscal (real)` – specifies the scaling factor applied to QM polarisabilities (cf. `qm_mm_polscal` `tinktep.config` directive). This keyword will be added automatically by TINKTEP, do not add it on your own. Default: 1.0.

`pol_emb_repulsive_mm_pot_cutoff (physical)` – specifies the cutoff (in length units) for the MM repulsive potential (cf. `rep` token in `tinktep.config` directives `onetep_perm_mpoles` and `onetep_induced_dipoles`). The value of the MM repulsive potential from an MM site will be assumed to be zero beyond this cutoff. This is used to minimise computational effort. The default is 10.0 bohr.

`pol_emb_mpole_exclusion_radius (physical)` – specifies the distance (in length units) around any multipole below which its Coulombic potential is masked or smeared (cf. `potential_coulombic_masked` and `potential_coulombic_smeared` arguments to `tinktep.config` directives `onetep_perm_mpoles` and `onetep_induced_dipoles`. This has no effect on Thole-damped multipole sets. The default is 0.25 bohr.

`pol_emb_fixed_charge (logical)` – if `T`, the MM force field will be assumed to be non-polarisable. If `F` (which is the default), the MM force field will be assumed to be polarisable. When the force field is polarisable and the QM* representation is used (`pol_emb_qmstar T`), `%block thole_polarisabilities` becomes mandatory. This keyword will be added automatically by TINKTEP, do not add it on your own.

`pol_emb_qmstar (logical)` – if `T`, the QM* representation will be employed for some or all QM-MM interactions. This is automatically set to `T` once polarisable embedding is activated. Only use `F` in the rare scenario, where the QM* representation will not be necessary (e.g. for non-polarisable force fields) and you want to elide `%block thole_polarisabilities`.

9.3 Uncommon and expert optional keywords

`pol_emb_thole_a (real)` – can be used to override the value of Thole's a parameter read from the `.prm` file. This value will be used *only* for QM/MM interactions, as TINKER will use the force-field value for MM/MM interactions. This keyword will be added automatically by TINKTEP if `qm_mm_thole_a` is specified in `tinktep.config`, do not add it on your own. Default: 0.39.

`pol_emb_smearing_a (physical)` – can be used to control the aggresiveness of Coulombic smearing (cf. `potential_coulombic_smeared` argument to `tinktep.config` directives `onetep_perm_mpoles` and `onetep_-induced_dipoles`. This has no effect on Thole-damped multipole sets. The default is 0.2 bohr.

`pol_emb_pairwise_polarisability (physical)` – see discussion of `potential_coulombic_smeared` argument to `tinktep.config` directives `onetep_perm_mpoles` and `onetep_induced_dipoles`. Default: 1.92618 bohr (yields the average AMOEBA09 polarisability). You should not have to use this keyword.

`pol_emb_repulsive_mm_pot_write (logical)` – if set to `T`, the repulsive MM potential will be written to a `.cube`/`.dx`/`.grd` file at every energy evaluation. Used for debugging purposes. Default: `F`.

`pol_emb_dbl_grid (logical)` – if set to `T` the polarisable embedding contribution to the NGWF gradient will be computed on the double grid. By default it is computed on the coarse grid, like in HFx. Technically, to prevent aliasing, the double grid version should be used, but it is unbearably inefficient and has not been optimised much, as I plan this to remain an experimental facility at most. To ensure consistency, `pol_emb_dbl_grid T` should only be used together with `swx_dbl_grid T`, which ensures NGWF-SWOP overlaps are calculated on the double grid too. This functionality has not been thoroughly tested. I strongly recommend leaving this at default (`F`).

`pol_emb_perm_scaling (real)` – all QM permanent multipoles will be scaled by this factor (default: 1.0, so no scaling). This can be used to test the effects of overpolarising/underpolarising QM/MM interactions.

9.4 Experimental vacuum-DMA functionality

This is an experimental, unpublished and hardly-tested functionality that allows having two different QM* descriptions – one for the vacuum density (i.e. in the absence of polarisable embedding), and one for the difference between the density with embedding present and without it ("polarisation density"). In principle this permits expanding the vacuum state density e.g. up to quadrupoles, and the difference e.g. only in dipoles, to mimic AMOEBA (by excluding "polarisable quadrupoles" and "fluctuating charges" in QM). Preliminary tests (2017) showed that this hardly matters, but it might be worth studying in more detail.

`pol_emb_vacuum_qmstar (logical)` – when set to `T`, it enables the vacuum-DMA functionality. Default: `F`, unless (all three simultaneously) `pol_emb_write_vacuum_restart F`, `pol_emb_vacuum_dma_max_l` is present and `pol_emb_qmstar T`.

`pol_emb_dma_min_l (integer)` – specifies the minimum angular momentum in the SW basis used in polarisable-embedding-DMA *for the polarisation density.* In most scenarios this will be equal to 0. This keyword does not affect properties-DMA (for which the corresponding value is always 0). This directly affects the quality of the QM* representation *for the polarisation density.* Setting `pol_emb_dma_min_l 1` and `pol_emb_dma_max_l 1` in particular will cause the polarisation density to be expanded only in terms of dipoles. Default: 0.

`pol_emb_vacuum_dma_min_l (integer)` – specifies the minimum angular momentum in the SW basis used in polarisable-embedding-DMA *for the in-vacuum density.* In most scenarios this will be equal to 0. This keyword does not affect properties-DMA (for which the corresponding value is always 0). This directly affects the quality of the QM* representation *for the in-vacuum density.* Default: 0.

`pol_emb_vacuum_dma_max_l (integer)` – specifies the maximum angular momentum in the SW basis used in polarisable-embedding-DMA *for the in-vacuum density.* In most scenarios this will be equal to 1 (up to dipoles) or 2 (up to quadrupoles). This keyword does not affect properties-DMA (for which `dma_max_l` should be used). This directly affects the quality of the QM* representation *for the in-vacuum density.* Default: -1 (indicating "unset").

`pol_emb_write_vacuum_restart (logical)` – if set to `T`, restart files (`.pol_emb_vac_tightbox_ngwfs` and `.pol_emb_kdenskern`) will be written every time the NGWF gradient is calculated. These restart files can be later used in the polarisation calculation. Default: `F`

9.5 Block keywords used in polarisable embedding

The following blocks are used to control polarisable embedding in ONETEP:

-
  ```
  %block thole_polarisabilities
  <QMatom1> <polarisability1>
  <QMatom2> <polarisability2>
  ...       ...
  <QMatomn> <polarisabilityn>
  %endblock thole_polarisabilities
  ```

  This block specifies Thole polarisabilities for all QM atoms, in units of bohr^3. Normally (i.e. if `qm_thole_polarisability` is specified in `tinktep.config`), this block will be automatically added by TINKTEP, which infers the QM polarisabilities from the `.prm` and `.tag` files. If you prefer to specify polarisabilities manually, add this block and omit `qm_thole_polarisability` from `tinktep.config`. For example for water TINKTEP would generate:

  ```
  %block thole_polarisabilities
  O 5.648355971436 ! Water O
  H 3.347173908999 ! Water H
  H 3.347173908999 ! Water H
  %endblock thole_polarisabilities
  ```

-
  ```
  %block mm_rep_params
  <MMspecies1> <A1> <ζ1>
  <MMspecies2> <A2> <ζ2>
  ...          ...
  <MMspeciesn> <An> <ζn>
  %endblock mm_rep_params
  ```

  This block specifies the MM repulsive potential parameters for all MM species in the system. The parameter A is the magnitude (in hartree), the parameter ζ is the inverse-width (in bohr^{-1}). For example for water, parameterised as in Ref. [4]:

  ```
  %block mm_rep_params
  H   35 2.400
  O  550 1.580
  %endblock mm_rep_params
  ```

10 Questions?

Questions should be directed to Jacek Dziedzic, `J.Dziedzic[-at-]soton.ac.uk`.

References

[1] A.J. Stone, GDMA: distributed multipoles from Gaussian98 wavefunctions (technical report), University of Cambridge (1998).

[2] A.J. Stone, Journal of Chemical Theory and Computation **6** 1128-1132 (2005).

[3] J. Dziedzic, Y. Mao, Y. Shao, J. Ponder, T. Head-Gordon, M. Head-Gordon and C.-K. Skylaris, J. Chem. Phys. **145** 12 124106 (2016).

[4] J. Dziedzic, T. Head-Gordon, M. Head-Gordon and C.-K. Skylaris, (in preparation) (2018).

[5] V. Vitale, J. Dziedzic, S.M.-M. Dubois, H. Fangohr and C.-K. Skylaris, Journal of Chemical Theory and Computation **11** 7 3321-3332 (2015).