Notes on QMMMW package

Changru Ma
SISSA, Trieste
and
Simone Piccinin
CNR-IOM and SISSA, Trieste

March 18, 2015

Contents

1 Introduction

QMMMW is an open-source GNU General Public License package for molecular dynamics within the frame work of Quantum Mechanics/Molecular Mechanics (QM/MM) hybrid scheme. The code was originally written by Riccardo Di Meo (SISSA - MS2 project) and Alessandro Laio. Changru Ma, Simone Piccinin and Alessandro Laio extended the original code to include the electrostatic coupling between the QM and MM regions. The present maintainers are Changru Ma and Simone Piccinin.

The QMMMW package operates as a wrapper that patches PWSCF code included in the Quantum ESPRESSO distribution (version 5.0.2) and LAMMPS Molecular Dynamics Simulator (version 30 Oct 2014 = stable version, SVN rev = 12671, Git tag = r12671). Please note that QMMMW is not compatible with all versions of Quantum ESPRESSO and LAMMPS.

The QMMMW package is written in Python, C and Fortran90 and designed in a modern way, taking advantage of an object-oriented structure. The package requires a recent version of Quantum ESPRESSO, a recent version of LAMMPS, Python newer than version 2.5, C and a Fortran-95 compiler.

With the current patches QMMMW allows to perform QM/MM simulations with

- Mechanical coupling (i.e. classical point charge based, QM and the MM system interact electrostatically only at the classical level.)
- Electrostatic coupling according to the scheme proposed by Laio, VandeVondele and Rothlisberger[1].

Current limitations (and planned developments):

- Only coded for orthorhombic cell.
- The interactions coupling QM and MM degrees of freedom can't be bonded. The QM/MM boundary can't cut through a chemical bond of a molecule.
- Long range electrostatic interaction is still excluded. The interaction between the QM system and the more distant MM atoms through a term explicitly coupling the multipole moments of quantum charge distribution with the classical point charges as described in Section III in Laio, VandeVondele and Rothlisberger[1] is not yet implemented.
- The QM region can only be modeled with PWSCF (i.e. the Car-Parrinello CP code can not be used to treat the QM part).

The Quantum ESPRESSO User Guide can be found here
http://www.quantum-espresso.org/user_guide/user_guide.html
and LAMMPS Manual can be found here
http://lammps.sandia.gov/doc/Manual.html
Further documentation regarding QMMMW can be found in doc/ directory.

1.1 Overview

The QM/MM approach was introduced by Warshel and Levitt in 1976[2]. In this hybrid model a portion of the system (QM) will be treated at the DFT level (Quantum ESPRESSO), while the remaining part of the system will be described using empirical force fields (`LAMMPS`). This approach is widely used for modeling for example large systems like biomolecules or systems in explicit solvent. The QM/MM method allows us to concentrate the computational effort on the crucial portion treated quantum mechanically while the MM region described by classical force fields provides an inexpensive description of the environment around the QM region.

1.2 Features of `QMMMW` implementations

Before performing a QM/MM simulation with `QMMMW`, these issues are worth considering:

- **QM/MM partitioning**
  - The boundary between QM and MM regions is fixed once it's defined. You have to decide which portion of your system is treated quantum mechanically and which portion is described by a classical force field.
  - Within the framework of `QMMMW`, the interactions coupling QM and MM degrees of freedom can't be bonded. The QM/MM boundary can't cut through a chemical bond of a molecule.
- **Choice of QM method**
  In principle, all features of PWSCF implemented in Quantum ESPRESSO are available for `QMMMW`. You have to choose your own pseudo-potentials, exchange-correlation functionals, *etc.*.
- **Choice of MM method**
  There are several force field be supported in `LAMMPS`, e.g., Lennard-Jones, Buckingham, Tersoff, REBO, AIREBO, ReaxFF, CHARMM, AMBER *etc.*. For a detail list, please have a look at http://lammps.sandia.gov/features.html#ff
- **Choice of electrostatic interaction between QM and MM subsystems**
  In `QMMMW`, we support two different schemes, mechanical coupling (MC) (i.e. classical point charge based, QM and the MM system interact electrostatically only at the classical level.) and electrostatic coupling (EC) according to the scheme proposed by Laio, VandeVondele and Rothlisberger[1]. With MC, you have to specify the charges on the atoms in the QM subsystem explicitly so that they are used to calculate the electrostatic interaction with MM subsystem. With EC, on the other hand, the charge density of the QM subsystem is calculated self-consistently by Quantum ESPRESSO.

2 Step-by-step QM/MM compilation with QMMMW

2.1 Compile QMMMW

In this section, we will show you how to obtain the codes and compile them. First of all, the QMMMW (qmmmw-1.0.tar.gz), QUANTUM ESPRESSO (espresso-5.0.2.tar.gz) and LAMMPS (lammps-stable.tar.gz) packages have to be downloaded from the following website and unzipped.

http://qe-forge.org/gf/project/qmmmw/frs/

To install QMMMW from source, you need first of all be able to compile and execute standard QUANTUM ESPRESSO and LAMMPS. Please make sure your QUANTUM ESPRESSO and LAMMPS work without any problems.

Second, you need Python newer than version 2.5 (at the time of writing, I am using Python 2.7.2). Then you need to run the `configure` script in QMMMW package indicating the paths for QMMMW, QUANTUM ESPRESSO and LAMMPS packages. Please define the variables $QMMMW_DIR (the directory you want to install the QMMMW package), $ESPRESSO_DIR (where you put QUANTUM ESPRESSO source code) and $LAMMPS_DIR (the directory of LAMMPS package).

```
./configure --prefix=$QMMMW_DIR ESPRESSO_DIR=$ESPRESSO_DIR LAMMPS_DIR=$LAMMPS_DIR
```

if the `configure` script terminates without any errors, you can patch QUANTUM ESPRESSO and LAMMPS now.

```
make patch
```

You should see

```
Patching make.sys.in... Patching PW/Makefile...Espresso correctly patched!
...
LAMMPS correctly patched!
```

If the patch fails with errors, it's likely that the version of QUANTUM ESPRESSO or LAMMPS is not compatible with QMMMW (at the time of writing, I am using the QUANTUM ESPRESSO 5.0.2 and LAMMPSversion 30 Oct 2014). Or maybe the variable $ESPRESSO_DIR and $LAMMPS_DIR have not been set properly. In the later case, try

```
ls $ESPRESSO_DIR
ls $LAMMPS_DIR
```

to see if these two variables are properly defined so the package knows whereQUANTUM ESPRESSO and LAMMPS source codes are.

In the next step, we need to compile and install the libraries and tools for the QMMMW package

```
make
make install
```

If the process terminates without errors, you have just completed the most difficult part in the installation. Congratulations!

2.2 Compile QUANTUM ESPRESSO

Then we need to compile QUANTUM ESPRESSO and LAMMPS. To compile the pw.x executable for QMMMW, just follow the normal procedure discussed in QUANTUM ESPRESSO User Guide[3].

```
cd $ESPRESSO_DIR
./configure
make pw
```

After the successful compilation, you will find the executable pw.x in $ESPRESSO_DIR/bin. Please remember, only compile QUANTUM ESPRESSO after it has been patched.

2.3 Compile LAMMPS

We prepared a Makefile.ms2 in $LAMMPS_DIR/src/MAKE for you as a starting point for the configuration file for LAMMPS. While in order to run the examples in $QMMMW/examples, we need to build a few standard packages first. Please follow the instructions:

```
cd $LAMMPS_DIR/src
make yes-kspace
make yes-molecule
make yes-rigid
```

to include the packages KSPACE and MOLECULE.

Next, since we just want to run LAMMPS on a single processor, we use the dummy MPI library provided in src/STUBS. To build this library, in src/STUBS directory, type

```
make
```

This should create a libmpi_stubs.a file suitable for linking to LAMMPS.

For a more detailed information about the what are these packages, how to build them in LAMMPS, please read LAMMPS Manual[4]. After the successful build of these packages, please have a look at the Makefile

$LAMMPS_DIR/src/MAKE/Makefile.ms2

modify it to satisfy your machine and compilers, make sure everything is correct, then just simply type

```
cd $LAMMPS_DIR/src
make ms2
```

After a successful compilation, you will find the executable lmp_ms2 in $LAMMPS_DIR/src.

At this point the installation of QMMMW, QUANTUM ESPRESSO and LAMMPS is finished. Every time you use the package, please don't forget to load the environment file in $QMMMW_DIR/bin/ms2.env.

```
source $QMMMW_DIR/bin/ms2.env
```

3 First worked example: 1 QM water molecule solvated in 31 MM water molecules

In this section, we will show you a very simple QM/MM simulation done with `QMMMW` package. The goal of this example is to explain you how to run a simulation using the `QMMMW` package. The simulation is composed of three processes, here we name as:

- **master**
  "master" is a `LAMMPS` process, it performs the time integration and handles the whole system.

- **slave**
  "slave" is also a `LAMMPS` process, but it's only used to compute the forces on the subsystem to be treated at quantum level.

- **qe**
  "qe" is a QUANTUM ESPRESSO process, it handles the same atoms in the "slave" process and deals with all the QM subsystem.

Beware, put the files for each process in separate directories, do not attempt to execute a `QMMMW` simulation with both "master" process and "slave" process in the same directory. And please run all these three simulations separately without `QMMMW` package (remove `fix 1 all ms2...` lines in `LAMMPS` input files and `MS2_enabled`, `MS2_handler` lines in QUANTUM ESPRESSO input file) in case you get errors, to find if there is any errors irrelevant to `QMMMW`.

3.1 Atom selection parser language

In this section, we will discuss about the atom selection parser language used in `QMMMW`. You can also read `$QMMMW/doc/atom_selection_syntax.txt`. In the "master" process, we have to specify which atoms are in the QM subsystem in a file `atoms.txt`. There are several ways to pick up QM atoms:

- TYPE type
  Atoms with type "type" in a `LAMMPS` data file containing information `LAMMPS` needs to run a simulation will be treated as QM atoms.

- TYPE $<, <=, =, >, >=$ type
  Same as above, but using an expression.

- INDEX index
  Select specified atom with index "index" starting from 1 as QM atom.

- index
  Shortcut for the former expression.

- INDEX $<, <=, =, >, >=$ index
  Select a subset of the indexes.

- index1 - index2
  All atoms in between "index1" and "index2" are QM atoms.

- ALL
  Select the whole system.

- NONE
  Select no atom.

The operators are logical, as: AND, OR (same as ","), XOR and NOT. We strongly recommend to use "()" braces with prodigality.

Here are some examples:

- TYPE 4
  All atoms of type 4.

- 1-100
  All atoms from 1 to 100, included.

- (INDEX < 100) AND (NOT TYPE 3)
  All atoms from 1 to 99, but not those of type 3.

- (TYPE < 4), TYPE 6, TYPE 12, 3
  All atoms of type 1, 2, 3, 6 and 12, plus the atom with index 3.

3.2 Mechanical coupling

Here is the first example, a mechanical coupling QM/MM simulation for 31 MM water molecules and 1 QM water molecule. The input files are in `$QMMMW_DIR/examples/32water_mc`. For the "master" process, the input file `water.in` looks like

```
units           real
neigh_modify    delay 0 every 1 check yes
atom_style      full
bond_style      harmonic
angle_style     harmonic
pair_style      lj/cut/coul/long 10.0
pair_modify     mix arithmetic
kspace_style    pppm 1e-4
special_bonds   amber

atom_modify sort 0 0

read_data       data.water
restart         100 restart1.qmmmw restart2.qmmmw

timestep        1.0
```

```
dump            1 all xyz 1 dump.xyz

thermo_style    multi
thermo          1

fix             1 all ms2 master atoms.txt shmem qmmm

fix             2 all nvt temp 300.0 300.0 70.0

fix             3 all shake 0.0001 20 10 b 1 a 1

run             2000
```

Here I will only explain the lines been used for `QMMMW`, for the description of these commands, please read `LAMMPS` Documentations [4].

First of all, `atom_modify sort 0 0` is required for `QMMMW` simulations in both "master" and "slave" processes. It turns off the spatial sorting or reordering of atoms within each processor's sub-domain to avoid the simulation scrambles the communication between others. Second, in order to enable the support for `QMMMW` in the "master" process, the lists for fixes should contain

```
fix             1 all ms2 master atoms.txt shmem qmmm
```

The option after `ms2`, `master` means the role of this process is a `LAMMPS` "master" process which contains the whole system. `atoms.txt` is the file name with the list of atoms that defines the QM subsystem. For the atom selection parser language, please read Section Atom selection parser language. In this example, `atoms.txt` has the following content:

```
25-27
```

This means, atoms with index 25, 26 and 27 are chosen as QM atoms. Then, `shmem` specifies the communicator used to communicate with the other processes. `shmem` uses the shared memory to transfer data and synchronize the codes. Since only the "master" process in each node will actually use the shared memory, this is viable on parallel setups, as long as all "mater" nodes are on the same machine. There are also other ways to transport, like `file` and `python`, please have a look at `$QMMMW_DIR/doc/ms2_transports.txt` for the details. `qmmm` is just a keyword identifies the code in the shared memory. It can be any word as long as the "master", "slave" and "qe" processes share the same one.

In `data.water`, we prepare a topological file for 31 TIP3P[5] water molecules and 1 QM water molecule (index 25-27) in a 9.865 $Å^3$ cubic box (Fig. 1).

The input file `water_single.in` for "slave" process is located in directory `slave`.

```
units           real
neigh_modify    delay 0 every 1 check yes
atom_style      full
```

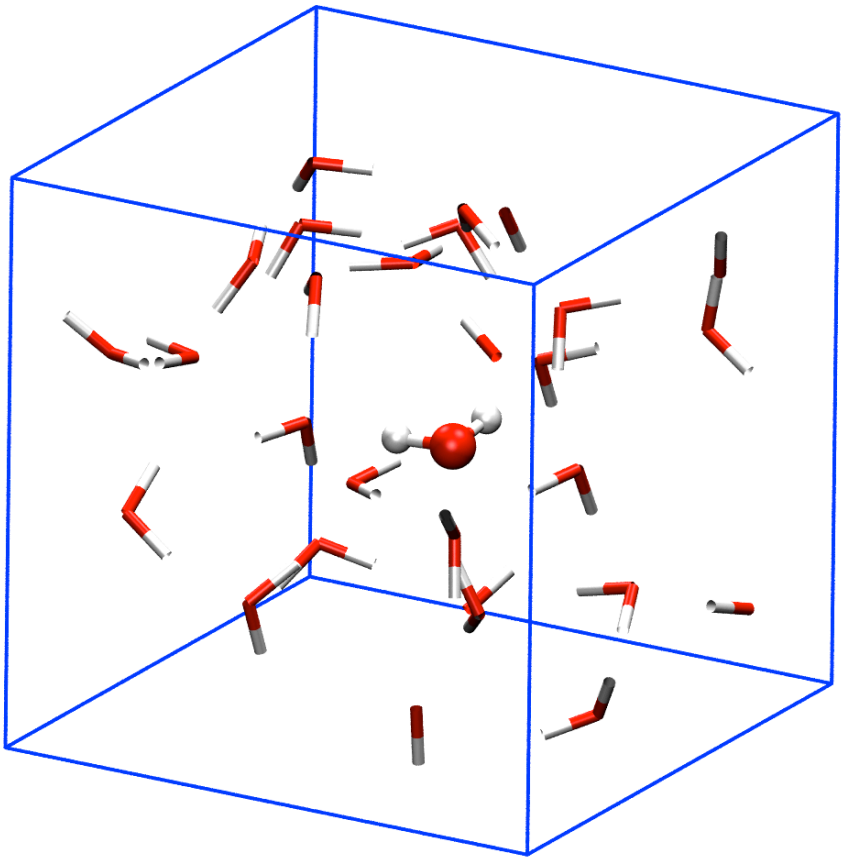


Figure 1: 31 MM water molecules and 1 QM water molecule. Red and white sticks represent MM oxygen and hydrogen atoms. Red and white balls represent QM oxygen and hydrogen atoms.

```
bond_style      harmonic
angle_style     harmonic
pair_style      lj/cut/coul/long 10.0
pair_modify     mix arithmetic
kspace_style    pppm 1e-4
special_bonds   amber

atom_modify sort 0 0

read_data       data.single_water

timestep        1.0

thermo_style    multi
thermo          1

fix             1 all ms2 slave shmem qmmm
run             2000
```

We change the option in `fix` from `master` to `slave` to identify this process as the LAMMPS executing the quantum subsystem. `shmem qmmm` is the same keyword as used on the "master" process.

In the data file `data.single_water`, we only put the information for the three atoms in the QM subsystem. It should contain only the appropriate sub-set of atoms, in the same order as in the "master" process. In the event that the subset atoms are two non contiguous groups, then the order within and between the two groups should be preserved. Please do not forget the information about bonds, angles and dihedrals for the QM atoms.

Now it's the turn of the QUANTUM ESPRESSO input file `water.in` in directory `qe`.

```
&CONTROL
title="QMMM"
calculation='md',
restart_mode = 'from_scratch',
tprnfor=.t.,
prefix='ms2',
pseudo_dir='pseudo'
nstep = 2000,
MS2_enabled = .true.,
MS2_handler = 'shmem qmmm',
/

&SYSTEM
ibrav = 1,
celldm(1) = 18.642155649244,
```

```
nat = 3,
ntyp = 2,
ecutwfc = 25.0 ,
ecutrho = 250.0 ,
/

&ELECTRONS
conv_thr  = 1.D-6,
/

&IONS
ion_positions = 'default'
/

ATOMIC_SPECIES
O 16.0000 O.pbe-van_bm.UPF
H 1.0000  H.pbe-van_ak.UPF

ATOMIC_POSITIONS angstrom
O 4.890520 4.325756 3.842052
H 3.893761 4.308535 3.950637
H 4.946166 4.514250 2.851741

K_POINTS gamma
```

The following lines in the name list `CONTROL` are added with respect to a normal Quantum ESPRESSO input.

```
MS2_enabled = .true.,
MS2_handler = 'shmem qmmm',
```

The first line enables the `QMMMW` package while the second line gives the transport method "shmem" and a keyword "qmmm" for it.

To run this simulation, it's mandatory to start the "master" process first, wait a few seconds for the initialization, then the "slave" process, again wait a few seconds, and the "qe" process is the last one to be executed. We prepared a example script `job.sh`.

```
#!/bin/bash

source $QMMMW_DIR/bin/ms2.env

cd master
$LAMMPS_DIR/src/lmp_ms2 < water.in > water.out &

sleep 5
```

```
cd ../slave
$LAMMPS_DIR/src/lmp_ms2 < water_single.in > water_single.out &

sleep 5

cd ../qe
$ESPRESSO_DIR/pw.x < water.in > water.out
```

Before using it, please identify the variables `$QMMMW_DIR`, `$LAMMPS_DIR` and `$ESPRESSO_DIR` as the directories where you install the `QMMMW` package, `LAMMPS` package and QUANTUM ESPRESSO package, respectively. Remember, load the environment `ms2.env` before running a simulation.

After all three processes are launched, these three codes will run at the same time. If for any reason one of the processes fails, the others will exit too and you will be able to inspect the causes of the problem. In the directory `reference`, we have done this example and prepared the output files as a reference.

To restart the QM/MM simulation, simply modify `master/water.in`, change `read_data data.water` to `read_restart restart1.qmmmw`, where `restart1.qmmmw` is the restart file in the "master" process.

In "master" process, first you should check the output file `water.out`, make sure MS2 module is working properly. If the following lines are in your output file, it means the "master" process is waiting for the "slave" and "qe" processes to begin in order to pass them the atom positions.

```
MS2 Module: init
MS2 Module: pre force
Setting up run ...
MS2 Module: setup
MS2 Module: post force
```

Between every MD step, you will see these two lines in the output. Forces calculated by "slave" and "qe" processes are sent back to "master" for the time integration of the molecular dynamics.

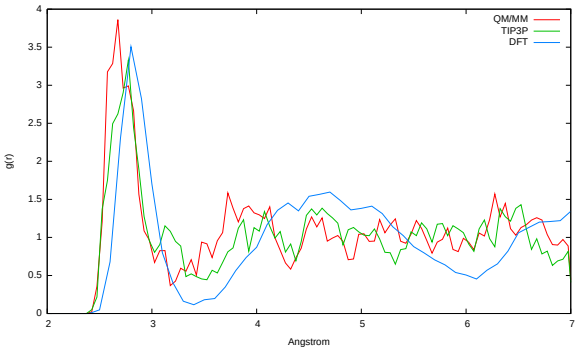
```
MS2 Module: pre force
MS2 Module: post force
```

In our example, positions of all atoms are dumped into `dump.xyz` every step. The first column is the index of atoms in `dump.xyz`, instead of atom names in a standard xyz file (1 for O and 2 for H in this example). We have a python script `xyz_tool.py` in directory `$QMMMW/tools` written by Riccardo Di Meo. It reads the `LAMMPS` format xyz and `LAMMPS` data file, looks for the atom names by reading the mass of the atoms, and then converts the `LAMMPS` xyz to a standard xyz file. This is the usage of the script:

```
xyz_tool.py dump.xyz -s -d data.water -o out.xyz
```

It reads `dump.xyz` and `data.water`, converts in the output file `out.xyz`. If `out.xyz` is already present, add `-f` to overwrite it.

Now you can visualize the animation, check if your results are reasonable or not. For example, here we check the radial distribution functions for quantum oxygen with respect to classical oxygen and classical hydrogen (Fig. 2). All simulations are performed in a box of 32 water molecules at normal density and at a temperature of 300 K for 2 ps only, therefor, the statistics is not sufficiently accurate.



(a) O-O

Figure 2: Quantum oxygen - classical oxygen pair correlation functions for a quantum water in a box of classical (TIP3P) water molecules compared with full classical TIP3P water and full quantum results.

3.3 Electrostatic coupling

The electrostatic coupling we implemented in this package the one proposed by Laio, VandeVondele and Rothlisberger[1]. Please read this article first. The polarization due to the MM atoms is described by a modified Coulombic potential

$$V(r) = \sum_{j \in NN} q_j v_j(|r - r_j|) + V_{lr}(r) \tag{1}$$

where NN is the set of MM atoms close to QM subsystem, with charge q_j, $V_{lr}(r)$ is the long range potential. The function v_j is taken as

$$v_j(r) = \frac{r_{cj}^n - r^n}{r_{cj}^{n+1} - r^{n+1}} \tag{2}$$

where $n = 4$ and r_{cj} is the covalent radius of the atom j. This potential is chosen to obtain the correct interaction properties and to prevent an unphysical escape of the electronic charge density to the MM atoms.

At the time of writing, all MM atoms are treated as short range atoms, the long range electrostatic iteration is not yet implemented, $V_{lr} = 0$.

There are a few differences between the configuration of a mechanical coupling (MC) simulation discussed above and an electrostatic one (EC). In this example, the

reference for the 1 QM water solvated in 31 MM water molecules electrostatic coupling test simulation is in directory `examples/32water_ec`.

You can copy input files from MC and modify them to become EC simulation. The first step to go from MC to EC is to set the charges on the QM atoms to zero in both "master" and "slave" processes. In this example, we know that QM atoms are of index 25, 26 and 27, therefore QM atoms in the `Atoms` section in the data file `data.water` in the "master" directory should change from

```
25 0 1 -0.83400 4.890520 4.325756 3.842052
26 0 2 0.41700 3.893761 4.308535 3.950637
27 0 2 0.41700 4.946166 4.514250 2.851741
```

to

```
25 0 1 0.00000 4.890520 4.325756 3.842052
26 0 2 0.00000 3.893761 4.308535 3.950637
27 0 2 0.00000 4.946166 4.514250 2.851741
```

Also, in the input file `water.in`, we need to replace the option in `fix` from `ms2` to `ms2ec`:

```
fix             1 all ms2 master atoms.txt shmem qmmm
```

to

```
fix             1 all ms2ec master atoms.txt shmem qmmm
```

Please keep in mind that, these two lines will load two completely separated libraries and plugins in `QMMMW`, source code for both can be found in `libms2` and `libms2ec`.

At this point the "master" process is correctly configured for the EC simulation and we move to the "slave" directory. We should change the charges in `data.single_water` to be all zero since here we only have atoms belong to QM subsystem. Also, the file `water_single.in` has to be modified a little bit in order to load the EC subroutines. Replace the line

```
fix             1 all ms2 slave shmem qmmm
```

with

```
fix             1 all ms2ec slave shmem qmmm
```

Since all charges in the "slave" process have been set to zero, the `pair_style` command in the input has to be changed from

```
pair_style      lj/cut/coul/long 10.0
```

to

```
pair_style      lj/cut/coul/cut 10.0
```

and turn off `kspace_style` by changing

```
kspace_style    pppm 1e-4
```

to

```
kspace_style    none
```

to exclude the additional damping factor applied to the Coulombic term used in conjunction with the `kspace_style` command and its `pppm`.

The last configuration we need to modify is in the "qe" directory. We will tell QUANTUM ESPRESSO that EC should be used. For this, we just need to change the line in `water.in`

```
MS2_handler = 'shmem qmmm',
```

to

```
MS2_handler = '$shmem qmmm',
```

The $ in the second line activates the EC section in the `ms2.f90` module and loads the `libms2base_shmem` plugin.

At this point the simulation can be started in the same way the MC one would be. You can find the covalent radius for all atoms in the output file `atomic_matching.txt` in "master" process. In "master" and "slave" processes, you can find MS2EC module is loaded instead of MS2 module and in "qe" process, the following line indicates EC is enabled.

```
XXXXXXXXXX EC SUPPORT ENABLED XXXXXXXXXX
```

4 Second worked example: alanine solvated with a bunch of water molecules

Alanine is an α-amino acid with the chemical formula $CH_3CH(NH_2)COOH$. In this example, the small bio-molecule is solvated in 514 water molecules (Fig. 3). Alanine is treated quantum mechanically and all the water molecules are treated classically. Input files and reference output files for both mechanical coupling simulation and electrostatic coupling simulation are located in `example/alanine_mc` and `alanine_ec`, respectively.

4.1 Mechanical coupling

First please have a look at the input configuration for "master" process, `examples/alanine_mc/maste`

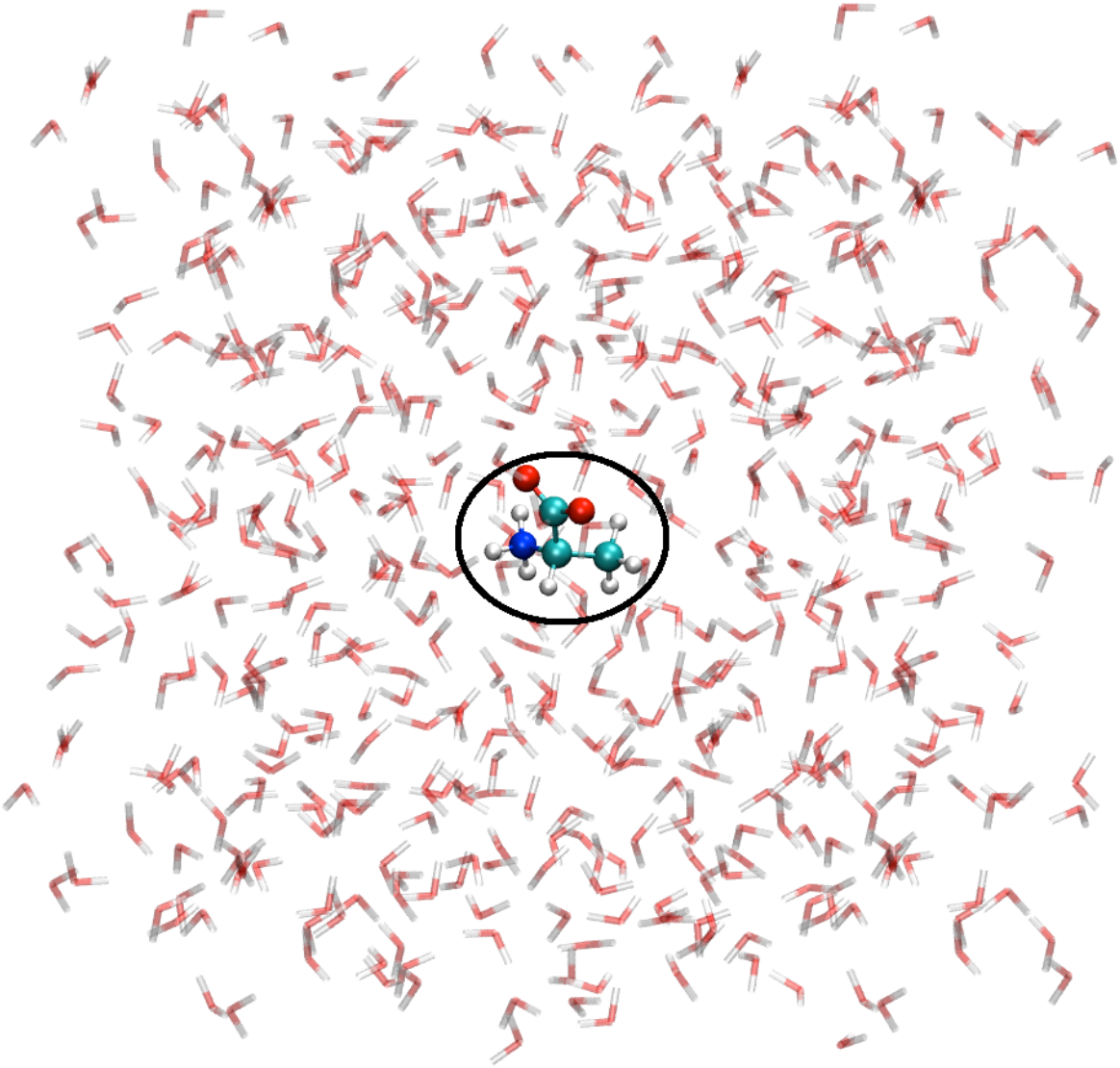


Figure 3: Alanine (inside the black cycle) solvated by 514 water molecules.

```
units           real
neigh_modify    delay 5
atom_style      full
bond_style      harmonic
angle_style     harmonic
dihedral_style  harmonic
pair_style      lj/cut/coul/long 10.0
pair_modify     mix arithmetic
kspace_style    pppm 1e-4

atom_modify sort 0 0

read_data       data.ala
special_bonds   amber

timestep        1

dump            1 all xyz 1 dump.alanine.xyz

thermo_style    multi
thermo          1

fix             1 all ms2 master atoms.txt shmem alanine
fix             2 all nvt temp 300.0 300.0 50.0
run             200
```

This input file is almost identical to the input for water example, apart from the fact that here we are using the keyword `alanine` instead of `qmmm` for the shared memory transportation. Also, in the file `atoms.txt` which gives the list of atoms in the QM subsystem, here we use

```
type <= 7
```

to set all atoms with type 1 to 7 as QM atoms.

In the "slave" and "qe" processes, please use the same keyword "alanine" for shared memory. All QM atoms have to be in the same order as they are in "master" process. Since these 13 QM atoms have 7 types in "master" and "slave" processes, we have to have the same number of types in "qe" process. You can name them C1, C2, H1, H2, H3, O and N like what we use in the example. And also, you may find that in the previous example, all three cells used for "master", "slave" and "qe" processes are identical, but here in the alanine example, the cell in "qe" process is different from that used in "master" and "slave". You are allowed to use smaller cells in QUANTUM ESPRESSO to save CPU time, just remember to check the convergence with respect to the cell dimension. As discussed above, only orthogonal MM cell is supported in `QMMMW` package.

4.2 Electrostatic coupling

Input files and references can be found at `examples/alanine_ec`.

Instructions for the impatient to move from MC to EC:

- Change the keyword `ms2` to `ms2ec` in the `fix` command in both "master" and "slave" processes.
- All charges on QM atoms should be zero in both "master" and "slave" processes.
- In "slave" process, use the proper `pair_style` and `space_style` since all atoms in this process are neutral.
- Change the value for `MS2_handler` in `CONTROL` name list in "qe" process from `shmem` to `$shmem`
- Execute "master", "slave" and "qe" processes in the same way.

References

[1] Laio, A and Vandevondele, J and Rothlisberger, U, A Hamiltonian electrostatic coupling scheme for hybrid Car–Parrinello molecular dynamics simulations, The Journal of Chemical Physics **116**, 6941 (2002)

[2] WARSHEL, A and LEVITT, M, Theoretical Studies of Enzymic Reactions - Dielectric, Electrostatic and Steric Stabilization of Carbonium-Ion in Reaction of Lysozyme, Journal of Molecular Biology **103**, 227 (1976)

[3] User's Guide for QUANTUM ESPRESSO: `espresso/Doc/`; http://www.quantum-espresso.org/user_guide/user_guide.html

[4] `LAMMPS` Documentation; http://lammps.sandia.gov/doc/Manual.htmll

[5] Price, Daniel J and Brooks, Charles L, A modified TIP3P water potential for simulation with Ewald summation, The Journal of Chemical Physics **121**, 10096 (2004)