i-PI 2.0: A universal Force Engine for Advanced Molecular Simulation

Yair Litman

i-PI 2.0: A universal Force Engine for Advanced Molecular Simulation

Venkat Kapil, **Mariana Rossi**, Ondrej Marsalek, Riccardo Petraglia, **Yair Litman**,Thomas Spura, Bingqing Cheng, Alice Cuzzocrea, Robert H. Meißner, David M. Wilkins, Przemyslaw Juda, Sébastien P. Bienvenue, Jan Kessler, Igor Poltavsky, Steven Vandenbrande, Jelle Wieme, Clemence Corminboeuf, Thomas D. Kühne, David E. Manolopoulos, Thomas E. Markland, Jeremy O. Richardson, Alexandre Tkatchenko, Gareth A. Tribello, Veronique Van Speybroeck , **Michele Ceriotti**

2

M. Ceriotti , et al., Com. Phys. Comm. 180, 1019 (2014)

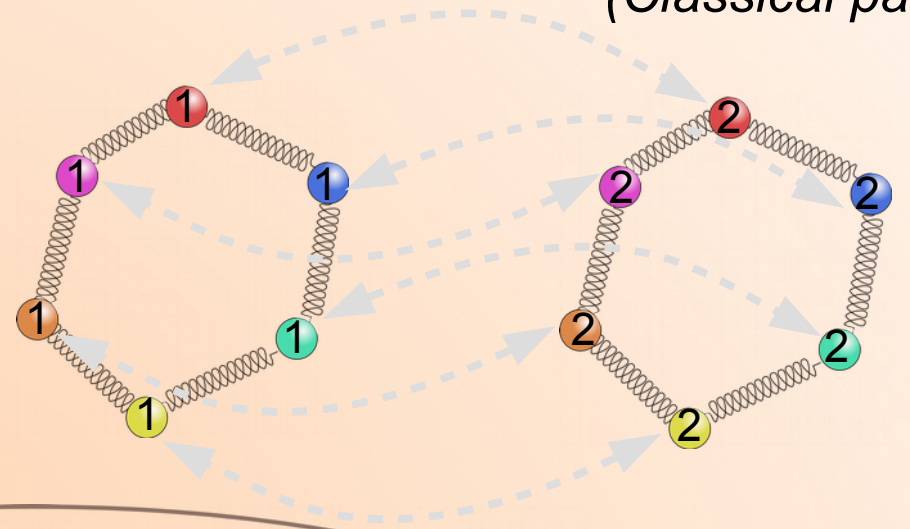
V. Kapil , et al., Com. Phys. Comm. (submitted 2018)

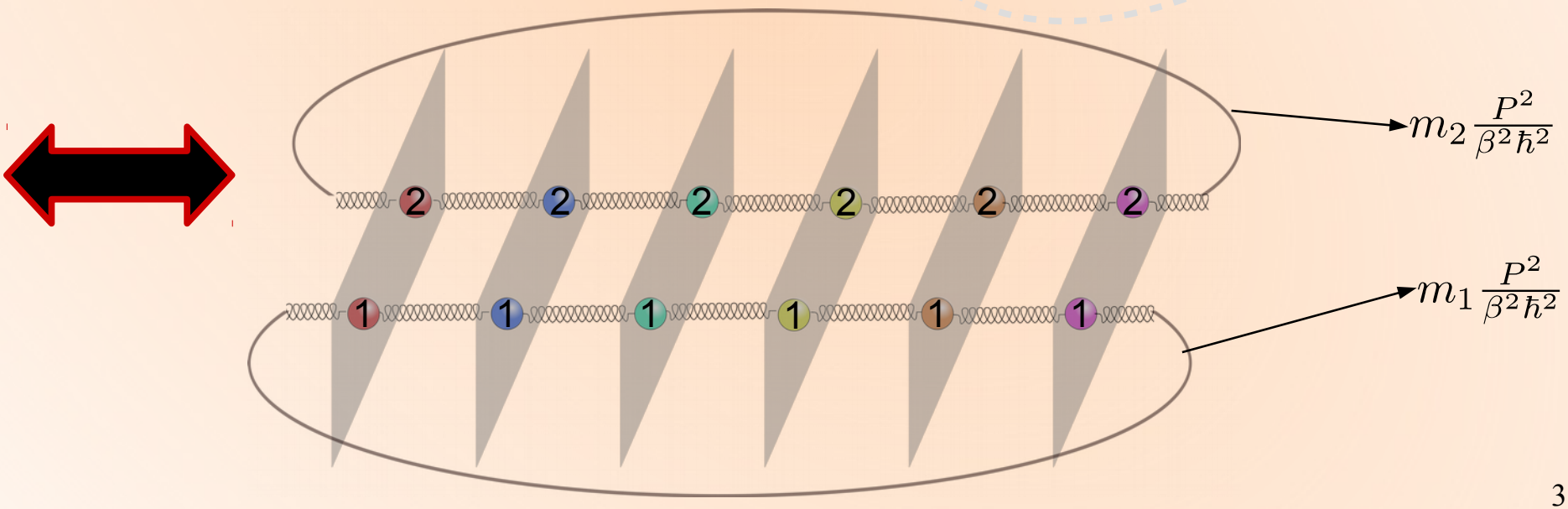
Path integral MD in one slide without equations

For a system of 2 distinguishable particles:

(Classical particles)

(Quantum particles)





Path Integral Molecular Dynamics (PIMD): Evolve the classical ring polymer at temperature equal P*T where,P is the number of replicas used in the simulation)

Infrastructure

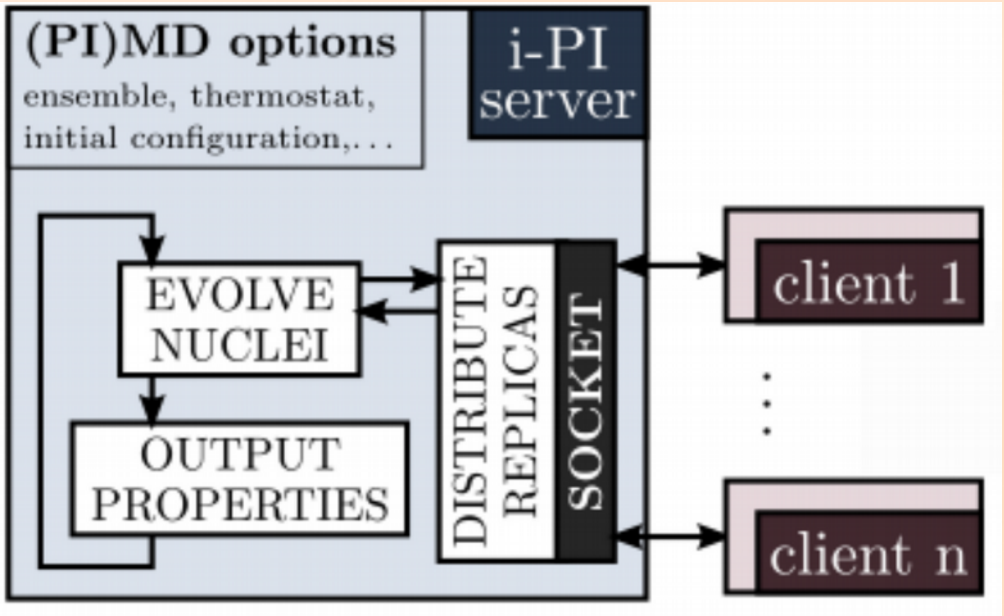
- It relays on the Born Oppenheimer (or ‘Adiabatic’) Approximation: the nuclear degrees of freedom are **decoupled** from the electronic ones.
- This do not imply that the nuclei are treated in a classical fashion.

Takes care of the electronic degrees of freedom

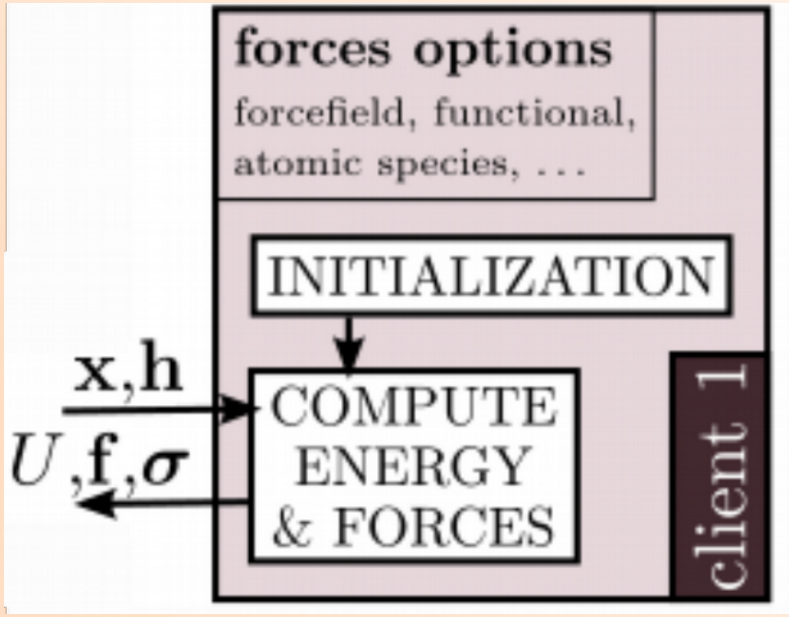
Takes care of the nuclei degrees of freedom

Infrastructure

Server (i-PI)



Client



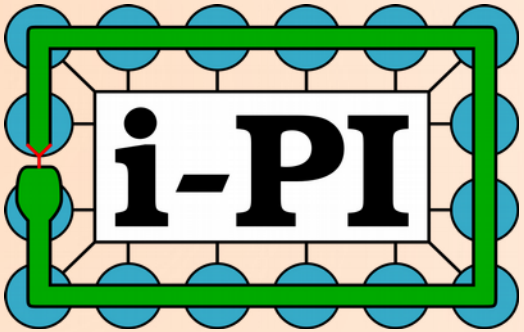
The communication is kept to a minimum. The **client is not restarted** between successive force evaluations, so that the overhead associated with initialization is avoided.

Picture adapted from: M. Ceriotti , et al., Com. Phys. Comm. 180, 1019 (2014)

Infrastructure

- It relays on the Born Oppenheimer (or 'Adiabatic') Approximation: the nuclear degrees of freedom are **decoupled** from the electronic ones.
- This do not imply that the nuclei are treated in a classical fashion.

Takes care of the electronic degrees of freedom



Takes care of the nuclei degrees of freedom

What were we able to do with i-PI original version?

- MD and Path Integral MD in the **NVE**, **NVT** and **NPT**

- **Thermostats**:

  ✔ stochastic velocity rescaling
  ✔ based on generalized Langevin equation (GLE):

    - the optimal sampling
    - quantum
    - δ
    - PIGLET (for Path Integral simulations)

- **Approximate quantum dynamics calculations**

  ✔ ring polymer molecular dynamics (RPMD)
  ✔ centroid molecular dynamics (CMD)

- Other “tools” for Path integral simulations such as **Ring polymer contraction**

What else we can do now?

On top of the previous capabilities, i-PI 2.0 has available several additional :

- **Optimizations**:
  - ✔ SD, CG, BFGS, LBFGS and BFGS-TRM.
  - ✔ Saddle point search (nichols)
- **MD:**
  - ✔ Multiple time stepping
  - ✔ Metadynamics through a PLUMED interface
  - ✔ Replica Exchange MD
- **Path Integral related techniques**
  - ✔ Reweighted and finite-difference fourth-order path integrals
  - ✔ Open path integrals and momentum distribution estimators
  - ✔ Spatially Localized Ring Polymer contraction
  - ✔ Ring Polymer Instanton
- **Others:**
  - ✔ Harmonic vibrations through finite differences
  - ✔ Self consistent phonons

Is it easy to get started?

- **Step 1**: Compile AIMS:

  - ex: make ipi.scalapack.mpi -f Makefile.ipi

- **Step 2**: Modify "control.in"

  - Add : use_pimd_wrapper <address> <port>
  - ex: use_pimd_wrapper localhost 40050
    communicate_pimd_wrapper dipole

- **Step 3**: Take an i-PI input from the example folder and adjust it.

- **Step 4**: Run your simulation:
  i) i-pi <input file>
  ii) aims.x

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
<total_time>  1.26000000e+04</total_time>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Example of i-PI input

```
<simulation verbosity='high'>
  <output prefix='simulation'>
    <properties stride='1' filename='out'>  [ step, time{picosecond}, conserved{electronvolt},
 temperature{kelvin}, potential{electronvolt}] </properties>
    <trajectory filename='pos' stride='20'> positions </trajectory>
    <checkpoint stride='200'/>
  </output>
  <total_steps>10000</total_steps>
  <prng>
    <seed>31415</seed>
  </prng>
  <ffsocket name='AIMS' mode='inet'>
    <address>localhost</address>
    <port>40050</port>
  </ffsocket>
  <system>
    <initialize nbeads='1'>
      <file mode='xyz'> init.xyz </file>
      <velocities mode='thermal' units='kelvin'> 300 </velocities>
      <cell mode='abc' units='angstrom'> [400.0, 400.0, 400.0] </cell>
    </initialize>
    <forces>
      <force forcefield='AIMS'> </force>
    </forces>
    <motion mode='dynamics'>
      <dynamics mode='nvt'>
        <timestep units='femtosecond'> 0.5 </timestep>
        <thermostat mode='langevin'>
          <tau units='femtosecond'> 100 </tau>
        </thermostat>
      </dynamics>
    </motion>
    <ensemble>
      <temperature units='kelvin'> 300 </temperature>
    </ensemble>
  </system>
</simulation>
```

Some problems/limitations should I know?

✗ The documentation, specially for the advance features, is scarce. (Even tough are examples available)

✗ If the time for a single force evaluation is below 1 second (example: Force Fields) the communications through the sockets is going to reduce the performance considerably.

✗ Change the number of atoms during the simulation (example: grand canonical ensemble) is not possible.

✗ Not solid state oriented code (q-point, symmetry, etc.)

✗ ...

Example of applications: MTS

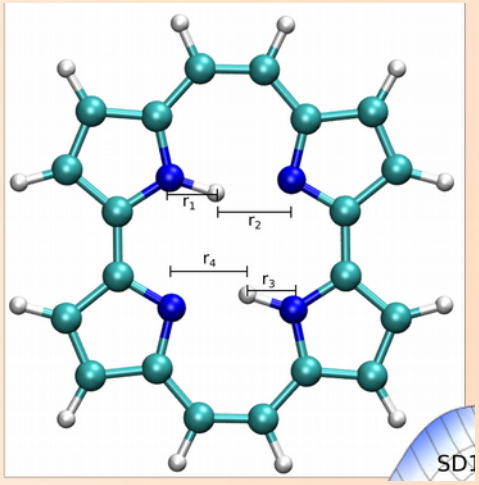
$$V(q) = V^{\mathrm{ref}}(q) + [V(q) - V^{\mathrm{ref}}(q)]$$

- The idea is to take V^{REF} as **cheap as possible**

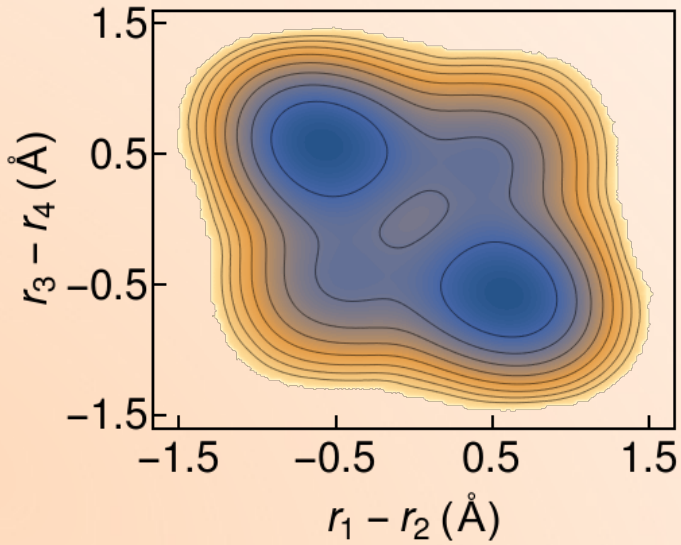
- If the difference [V - V^{REF}] is slow varying in time, we can used a longer time step to integrate it, i.e. we don't have to evaluate the expense V every step.

- V and V^{REF} could be: (FF, HYBRID) , (GGA,MP2), (GGA(light), GGA(tight), (GGA,CC),etc

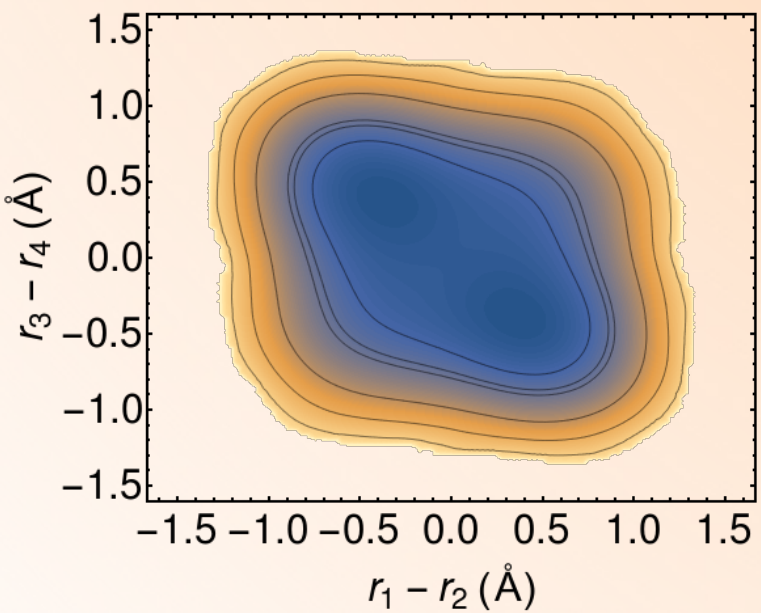
Example of applications: RPC



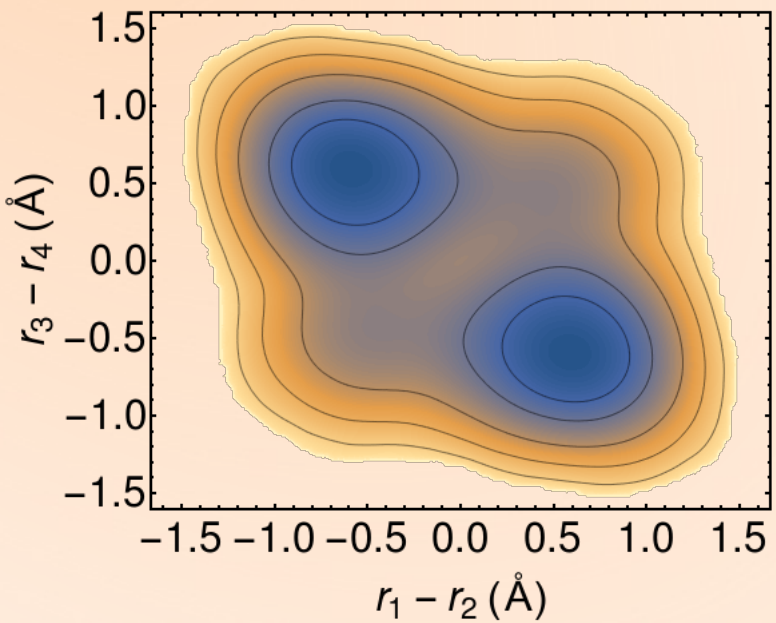
B3LYP



MBD



RPC(B3LYP-MBD)



Comparison with ASE

	i-PI	ASE
Compatibility	☺	☺
Optimization	☺	☺
Interactive	✗	☺
MD	☺	☺
PIMD	☺	✗
High-throughput	✗	?

Final remarks

- Universal force engine: it can connect with several codes. Therefore, any implementation is **automatically available** for all of them.

- It contains several well established and last generation constantly maintained to speed up MD and PIMD simulations.

- 2 main developers here in our department.

- Easy to get started.

- The code is more actively maintained than the MD functionality in FHI-AIMS


Thank You!