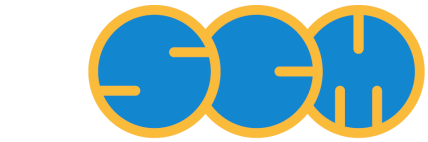

Scientific Computing & Modelling

PyMD Manual

ADF Program System
Release 2012

Scientific Computing & Modelling NV
Vrije Universiteit, Theoretical Chemistry
De Boelelaan 1083; 1081 HV Amsterdam; The Netherlands
WWW: www.scm.com
E-mail: support@scm.com

PyMD: A python library for flexible multi-scale molecular dynamics simulation

Table of Contents

Basic philosophy

We present a flexible python library for molecular dynamics, specialized in multi-scale simulations in a broad sense. At its core, the library interfaces the Atomistic Simulation Environment (ASE) molecular dynamics modules with a wide range of molecular mechanics and electronic structure codes. As such, it allows simple dynamics using forces computed with any energy/gradient evaluator provided by the ADF package.

Additionally, PyMD allows the partitioning of a system into regions described at different resolution, with the aim of running multi-scale (hybrid) force calculations. Besides the traditional, rigid, multi-scale partitioning, PyMD includes different schemes for Adaptive Multi-scale Molecular Dynamics. Such simulations allow the resolution of a particle to change according to its distance from a predefined active site, which is a necessity for successful multi-scale description of diffusive systems like chemical reactions in solution.

Finally, the library couples the dynamics to rare events techniques, either implemented in PyMD itself, or accessible through an interface with the PLUMED library for free energy calculations, opening the possibility for evaluation on time-scales beyond the reach of standard molecular dynamics simulations.

1. Introduction

PyMD is a python package providing molecular dynamics (MD) simulations using the energy evaluation methods made available by the ADF suite. A set of example scripts can be found in the examples/scmlib directory of a standard ADF installation.

PyMD can be accessed interactively by running `startpython`, followed by a standard python import command for the package `scm.pymd`. The python help function can be used to obtain detailed documentation about all PyMD classes. In the following example, an inquiry of one class (the MDMolecule class) can be performed.

```
$ startpython
>>> from scm import pymd
>>> help(pymd.MDMolecule)
```

2. Molecular Dynamics

PyMD defines the molecular system under study through the **MDMolecule** class: an instantiation of this class holds all information about the molecular system to be simulated, such as coordinates, topology, and force field parameters (if needed). An **MDMolecule** object can be initialized from a PDB or XYZ file, by specifying its path at object creation.

An interface to energy evaluators is provided by specialized **ForceJob** classes, acting as wrappers around the ADF suite of programs. A **ForceJob** requires an **MDMolecule** object to be specified at creation. The resulting **ForceJob** object can either be used directly by the Atomistic Simulation Environment (ASE) [1] library as a calculator object (see examples/scmlib/ASE_emt_h2o) or with the **ASEMDPropagator** class, which provides methods for running an MD time step using ASE classes. Internally, the propagator sets up the required ASE objects, passes the **ForceJob** object to them, and retrieves the new positions and velocities. An additional protagonist, an **MDManager** class instance, coordinates the MD simulation by running the MD steps with the **ASEMDPropagator** object and writing trajectory information to file.

During creation of an **MDManager** object, a directory ‘QMMD’ is created, which contains a file TRAJEC00.DCD holding the geometries along the trajectory, a file FTRAJEC00.DCD holding the forces along the trajectory, and finally a file ENERGY00.dat holding the potential and kinetic energy, as well as the temperature throughout the evaluation. To extract the geometries from the trajectory file, the **DCDFile** class is available, providing methods to read and write geometries to and from a trajectory file in DCD format. The **MDManager** is also responsible for handling restart of a previous MD evaluation: if a 'QMMD' directory is already present at script invocation, the new output files will be assigned the number subsequent to the highest numbered files in that directory. In addition, provided the previous run terminated normally, the restart will continue from the final geometry and velocity of the previous run.

The ADF package contains different electronic structure methods of varying degrees of accuracy and speed. The best-known methods are the ADF Density Functional Theory (DFT) code itself, and the BAND DFT code for periodic systems. PyMD provides an interface toward both programs. For the interface with ADF, PyMD makes use of classes from PyADF [2], a scripting framework for efficient quantum chemistry calculations. In addition to ADF and BAND, several semi-empirical methods are included in the ADF suite, such as DFTB and the NDDO type schemes available in the MOPAC package [3]. The ADF suite also provides classical mechanics methods, such as the reactive force field ReaxFF and the simple force field UFF. Interfaces to all of these methods are available in PyMD. A simple example of a python script for MD using the **UFFForceJob** class for UFF calculations can be found in the examples directory, under `examples/scmlib/pymd_uff_h2o`.

To increase the flexibility of PyMD, an interface towards force calculations using the NAMD2.8 classical molecular dynamics package is provided (`examples/scmlib/namd_h2o`). NAMD2.8 is not distributed with the ADF suite, but it is available from a third party to be downloaded and installed (http://www.ks.uiuc.edu/Development/Download/download.cgi).

3. Multi-scale Molecular Dynamics

The design of the **ForceJob** class allows for flexible extension of its behavior, while at the same time keeping the client code unaware of its nature: it can either act as a simple wrapper for ADF programs, or it can be a more complex orchestrating class, combining simpler **ForceJob** classes to implement multi-scale strategies. One application of this extensible design can be found in the **QMMMForceJob** object, which combines a QM and an MM method in an IMOMM-type scheme (mechanical embedding). The **QMMMForceJob** object is assigned two other **ForceJob** objects, the first representing the high-resolution calculation (QM), while the other represents the low resolution (MM). Both **ForceJob** objects contain an **MDMolecule** object for the full molecular system. The selection of the QM-region is handled by the **QMMMForceJob**, which contains the information about the part of the molecule that constitutes the QM region. When forces are requested from the **QMMMForceJob**, the following behavior is orchestrated: first, a MM force calculation is performed on the full system; then, the QM-region is selected, a QM calculation is executed solely for that region, and energy and forces are added to those from the full system MM calculation. Finally, an MM calculation is computed for the small QM-region, and the energy and forces are subtracted, yielding the final result, returned to the invoker. In symbols:

EQM/MM(Full) = EMM(Full) + EQM(QMRegion) – EMM(QMRegion)

The **QMMMForceJob** handles periodic boundary conditions if the low-level (MM) method supports this feature (i.e. NAMD). Whether the periodic interaction of the QM region with itself is handled at high or low resolution depends on the method used for the QM calculation. An example of QM/MM MD calculations can be found in the examples directory `examples/scmlib/qmmm_dftbUFF_2h2o`. The **QMMMForceJob** allows the use of link atoms when the QM boundary cuts through covalent bonds. However, this feature comes at the price of an increased script complexity. An example of a QMMM link-atom MD simulation is provided in the examples directory, under `examples/scmlib/qmmm_linkatom_dftbNAMD_glutamate`.

For more complex multi-scale calculations the **HybridForceJob** class can be used. This class allows the combination of a large set of different **ForceJob**s, each of them describing either the same, or different molecular systems. Each **ForceJob** can either involve a calculation on the full **MDMolecule** object it contains, or restricted to a specified region of the corresponding molecule. The forces from each contributing **ForceJob** can either be added or subtracted from the total force according to user preference, as specified at construction of the **HybridForceJob** object.

In order to perform QM/MM simulations on chemical reactivity in solution, it is important that the description of the solvent molecules can change on the fly, as the molecules move towards or away from the reactive region. To facilitate this, an **AdaptiveQMMMForceJob** class is available to provide adaptive QM/MM simulations using several available schemes, as described by Bulo et al.[4] In these schemes, the description of the diffusing molecules changes gradually from QM to MM and vice versa, based on the distance of those molecules to a predefined reactive site. Various schemes are available for assigning the QM and MM character of the molecules. The class contains a **QMMMForceJob** object, as well as a partitioning object that assigns the partial QM and MM character to the molecules. An examples python script for such an adaptive QM/MM simulation, using the DAS [4] method, is provided in the examples directory, `examples/scmlib/adqmmm_mopacscmUFF_h2o`.

4. Biased Molecular Dynamics

Constraints can be added to a simulation using the derived **ForceJob** class **WallJob**. The constraint is in the form of a large one-dimensional Gaussian on the potential energy surface, along a predefined Collective Variable (CV). Examples of CV's are the distance between two atoms, the coordination number of two atoms, but also more complex quantities such as the minimum distance between two sets of atoms, or the distance of an atom to a hydroxide ion. The Collective Variables can be specified through the **CollectiveVariable** class. Derived **CollectiveVariable** classes are available to specify sums or multiples of other **CollectiveVariable** objects.

Regular MD calculations are limited in the time-scales achievable with current hardware. The order of such time-scales is much smaller than what is required for chemical reactions. To overcome this problem, two rare-events methods have been implemented directly into the library: metadynamics [5] and umbrella sampling [6]. Both these methods involve biasing the simulations along a CV. An example of a metadynamics input can be found in the examples directory in `examples/scmlib/metadynamics_emt_h2o`.

For a wider range of rare-events methods, PyMD also offers an interface with the PLUMED library for free energy calculations[7]. To use this, a PLUMED input file is required, and for this we refer to the PLUMED manual. An example of a PyMD input script using PLUMED can be found in the examples directory in `examples/scmlib/plumed_emt_h2o`.

References

1. S. R. Bahn, K. W. Jacobsen, *An object-oriented scripting interface to a legacy electronic structure code.* Comput. Sci. Engin. **4**, 56-66 (2002)

2. C. R. Jacob, S. M. Beyhan, R. E. Bulo, A. Severo Pereira Gomes, A. W. Gotz, K. Kiewisch, J. Sikkema, L. Visscher, *PyADF - A scripting framework for multiscale quantum chemistry.* J. Comput. Chem. **32**, 2328-2338 (2011)

3. J. J. P. Stewart, *Optimization of parameters for semiempirical methods IV: extension of MNDO, AM1, and PM3 to more main group elements.*, J. Mol. Model. **10**, 155-164 (2004)

4. R. E. Bulo, B. Ensing, J. Sikkema, L. Visscher, *Toward a Practical Method for Adaptive QM/MM Simulations.* J. Chem. Theory Comput. **5**, 2212-2221 (2009)

5. A. Laio, M. Parrinello, *Escaping free-energy minima.* Proc. Natl. Acad. Sci. USA., **99**, 12562-12566 (2002)

6. B. Roux, *The calculation of the potential of mean force using computer simulations.* Comput. Phys. Commun. **91**, 275-282 (1995)

7. M. Bonomi, D. Branduardi, G. Bussi, C. Camilloni, D. Provasi, P. Raiteri, D. Donadio, F. Marinelli, F. Pietrucci, R. A. Broglia, M. Parrinello, *PLUMED: A portable plugin for free-energy calculations with molecular dynamics.* Comp. Phys. Comm. **180**, 1961-1972 (2009)