hoomd$_{f}lowwsDocumentation$

Release 0.1

Matthew Spellings

Feb 13, 2020

Contents:

`hoomd_flowws` is an in-development set of modules to create reusable scientific workflows using hoomd-blue. While the python API of hoomd-blue holds enormous possibility for scriptability (including making projects like this possible in the first place), this flexibility can also lead to poorly-structured, rigid script workflows if not carefully managed. The aim of this project is to formulate a set of robust, modular individual components that can be composed to perform most common workflows.

`hoomd-flowws` is being developed in conjunction with flowws.

CHAPTER 1

Installation

Install `hoomd_flowws` from PyPI:

```
pip install hoomd_flowws
```

Alternatively, install from source:

```
pip install git+https://github.com/klarh/hoomd_flowws.git#egg=hoomd_flowws
```

CHAPTER 2

Documentation

Browse more detailed documentation online or build the sphinx documentation from source:

```
git clone https://github.com/klarh/hoomd_flowws
cd hoomd_flowws/doc
pip install -r requirements.txt
make html
```

2.1 Modules

class hoomd_flowws.**Init**(***kwargs*)
Initialize a system

Currently simply places points on a simple cubic lattice.

Parameters

- **number** – Number of particles to simulate
- **mass_scale** – Scaling factor for mass of all particles
- **type_ratios** – Prevalence (ratio) of each particle type

run(*scope*, *storage*)
Run the contents of this stage

class hoomd_flowws.**Interaction**(***kwargs*)
Specify a new interaction potential to include in future MD stages

Parameters

- **reset** – Clear previously-defined interactions beforehand
- **type** – Interaction class name
- **global_params** – Global parameters of the interaction
- **pair_params** – Type pair-based parameters of the interaction

classmethod register_interaction(*interaction*, **names*)
Convenience method to bind one or more names to a force-generation function

run(*scope*, *storage*)
Registers this object to provide a force compute in future MD stages

class hoomd_flowws.**ShapeDefinition**(***kwargs*)
Define per-type shapes for future stages to utilize

Shape information is used for visualization, packing fraction calculations, and pair force/HPMC integrator configuration.

Shapes consist of a base type, any parameters of the shape, and modifications. For example:

```
# regular polygon with 4 vertices (square)
shape = dict(type='regular_ngon', n=4,
             modifications=[dict(type='scale', factor=2)])
# rounded tetrahedron
shape = dict(type='tetrahedron',
             modifications=[dict(type='round', radius=0.5)])
ShapeDefinition(shape_arguments=[shape])
```

Parameters **shape_arguments** – List of per-shape specifications and modifiers.

run(*scope*, *storage*)
Run the contents of this stage

class hoomd_flowws.**DEMInteraction**(***kwargs*)
Specify that DEM interactions should be included in future MD stages

Parameters

- **reset** – Clear previously-defined DEM interactions beforehand
- **type** – Interaction class name

run(*scope*, *storage*)
Registers this object to provide a DEM force compute in future MD stages

class hoomd_flowws.**Run**(***kwargs*)
Run for a given number of timesteps using MD

Parameters

- **steps** – Number of timesteps to run
- **timestep_size** – Timestep size
- **integrator** – Integrator type
- **temperature** – Temperature for isothermal simulations
- **ramp_t_to** – Ramp temperature to the given value over time during this stage
- **tau_t** – Thermostat time constant for isothermal simulations
- **pressure** – Pressure for isobaric simulations
- **tau_p** – Barostat time constant for isobaric simulations
- **bd_seed** – Random number seed for Brownian/Langevin thermostats
- **zero_momentum** – Period for zeroing the momentum of the system (default: disabled)
- **backup_period** – Period for dumping a backup file

- **dump_period** – Period for dumping a trajectory file
- **trajectory_quantities** – Additional trajectory quantities to dump
- **dump_quantities** – Dump period and comma-separated list of quantities to record (i.e. temperature,pressure)
- **expand_by** – Expand each dimension of the box by this ratio during this stage
- **compress_to** – Compress to the given packing fraction during this stage (overrides expand_by)

run(*scope*, *storage*)
Run the contents of this stage

class hoomd_flowws.**RunHPMC**(***kwargs*)
Run for some number of steps using HPMC

Parameters

- **steps** – Number of timesteps to run
- **integrator** – Integrator type
- **pressure** – Pressure for isobaric simulations
- **backup_period** – Period for dumping a backup file
- **dump_period** – Period for dumping a trajectory file
- **expand_by** – Expand each dimension of the box by this ratio during this stage
- **compress_to** – Compress to the given packing fraction during this stage (overrides expand_by)
- **integrator_seed** – Random number seed for integration method
- **tune** – Tune move distances to achieve the target acceptance ratio after updating a given number of epochs and running the given number of steps at each epoch
- **box_move_aspect** – Move distance and weight for box aspect ratio moves
- **box_move_length** – Move distance and weight for box length moves
- **box_move_ln_volume** – Move distance and weight for box log-volume moves
- **box_move_shear** – Move distance and weight for box shear moves
- **box_move_volume** – Move distance and weight for box volume moves

2.2 Indices and tables

- genindex
- modindex
- search

Index