

NAME

OpenMMPerformMinimization.py - Perform an energy minimization

SYNOPSIS

```
OpenMMPerformMDSimulation.py [--forcefieldParams <Name,Value,...>] [--freezeAtoms <yes or no>] [
--freezeAtomsParams <Name,Value,...>] [--integratorParams <Name,Value,...>] [--outputParams
<Name,Value,...>] [--outfilePrefix <text>] [--overwrite] [--platform <text>] [--platformParams
<Name,Value,...>] [--restraintAtoms <yes or no>] [--restraintAtomsParams <Name,Value,...>] [
--restraintSpringConstant <number>] [--simulationParams <Name,Value,...>] [--smallMolFile
<SmallMolFile>] [--smallMolID <text>] [--systemParams <Name,Value,...>] [--waterBox <yes or no>] [
--waterBoxParams <Name,Value,...>] [-w <dir>] -i <infile>
```

OpenMMPerformMDSimulation.py -h | --help | -e | --examples

DESCRIPTION

Perform energy minimization for a macromolecule or a macromolecule in a complex with small molecule. You may optionally add a water box and freeze/restraint atoms to your system before minimization.

The input file must contain a macromolecule already prepared for simulation. The preparation of the macromolecule for a simulation generally involves the following: tasks: Identification and replacement non-standard residues; Addition of missing residues; Addition of missing heavy atoms; Addition of missing hydrogens; Addition of a water box which is optional.

In addition, the small molecule input file must contain a molecule already prepared for simulation. It must contain appropriate 3D coordinates relative to the macromolecule along with no missing hydrogens.

The supported macromolecule input file formats are: PDB (.pdb) and CIF (.cif)

The supported small molecule input file format are : SD (.sdf, .sd)

Possible outfile prefixes:

```
<InfieRoot>
<InfieRoot>_Solvated
<InfieRoot>_<SmallMolFileRoot>_Complex
<InfieRoot>_<SmallMolFileRoot>_Complex_Solvated
```

Possible output files:

```
<OutfilePrefix>_Initial.<pdb or cif>
<OutfilePrefix>_Minimized.<pdb or cif>
<OutfilePrefix>_Minimization.csv
```

OPTIONS

-e, --examples

Print examples.

-f, --forcefieldParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for biopolymer, water, and small molecule forcefields.

The supported parameter names along with their default values are shown below:

```
biopolymer, amber14-all.xml [ Possible values: Any Valid value ]
smallMolecule, openff-2.2.1 [ Possible values: Any Valid value ]
water, auto [ Possible values: Any Valid value ]
additional, none [ Possible values: Space delimited list of any
valid value ]
```

Possible biopolymer forcefield values:

```
amber14-all.xml, amber99sb.xml, amber99sbildn.xml, amber03.xml,
amber10.xml
charmm36.xml, charmm_polar_2019.xml
amoeba2018.xml
```

Possible small molecule forcefield values:

```
openff-2.2.1, openff-2.0.0, openff-1.3.1, openff-1.2.1,
openff-1.1.1, openff-1.1.0, ...
```

smirnoff99Frosst-1.1.0, smirnoff99Frosst-1.0.9, ...
gaff-2.11, gaff-2.1, gaff-1.81, gaff-1.8, gaff-1.4, ...

The default water forcefield value is dependent on the type of the biopolymer forcefield as shown below:

Amber: amber14/tip3pfb.xml
CHARMM: charmm36/water.xml or None for charmm_polar_2019.xml
Amoeba: None (Explicit)

Possible water forcefield values:

amber14/tip3p.xml, amber14/tip3pfb.xml, amber14/spce.xml,
amber14/tip4pew.xml, amber14/tip4pfb.xml,
charmm36/water.xml, charmm36/tip3p-pme-b.xml,
charmm36/tip3p-pme-f.xml, charmm36/spce.xml,
charmm36/tip4pew.xml, charmm36/tip4p2005.xml,
charmm36/tip5p.xml, charmm36/tip5pew.xml,
implicit/obc2.xml, implicit/GBn.xml, implicit/GBn2.xml,
amoeba2018_gk.xml (Implicit water)
None (Explicit water for amoeba)

The additional forcefield value is a space delimited list of any valid forcefield values and is passed on to the OpenMMForcefields SystemGenerator along with the specified forcefield values for biopolymer, water, and small molecule. Possible additional forcefield values are:

amber14/DNA.OL15.xml amber14/RNA.OL3.xml
amber14/lipid17.xml amber14/GLYCAM_06j-1.xml
... ..

You may specify any valid forcefield names supported by OpenMM. No explicit validation is performed.

--freezeAtoms <yes or no> [default: no]

Freeze atoms during energy minimization. The specified atoms are kept completely fixed by setting their masses to zero. Their positions do not change during energy minimization.

--freezeAtomsParams <Name,Value,...>

A comma delimited list of parameter name and value pairs for freezing atoms during energy minimization. You must specify these parameters for 'yes' value of '--freezeAtoms' option.

The supported parameter names along with their default values are shown below:

selection, none [Possible values: CAlphaProtein, Ions, Ligand,
Protein, Residues, or Water]
selectionSpec, auto [Possible values: A space delimited list of
residue names]
negate, no [Possible values: yes or no]

A brief description of parameters is provided below:

selection: Atom selection to freeze.
selectionSpec: A space delimited list of residue names for
selecting atoms to freeze. You must specify its value during
'Ligand' and 'Protein' value for 'selection'. The default values
are automatically set for 'CAAlphaProtein', 'Ions', 'Protein',
and 'Water' values of 'selection' as shown below:

CAAlphaProtein: List of standard protein residues from pdbfixer
for selecting CAAlpha atoms.
Ions: Li Na K Rb Cs Cl Br F I
Water: HOH
Protein: List of standard protein residues from pdbfixer.

negate: Negate atom selection match to select atoms for freezing.

In addition, you may specify an explicit space delimited list of residue names using 'selectionSpec' for any 'selection'. The specified residue names are appended to the appropriate default values during the selection of atoms for freezing.

-h, --help

Print this help message.

-i, --infile <infile>

Input file name containing a macromolecule.

--integratorParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for integrator to setup the system for local energy minimization. No MD simulation is performed.

The supported parameter names along with their default values are shown below:

temperature, 300.0 [Units: kelvin]

--outfilePrefix <text> [default: auto]

File prefix for generating the names of output files. The default value depends on the names of input files for macromolecule and small molecule along with the type of statistical ensemble and the nature of the solvation.

The possible values for outfile prefix are shown below:

<InfileRoot>
 <InfileRoot>_Solvated
 <InfileRoot>_<SmallMolFileRoot>_Complex
 <InfileRoot>_<SmallMolFileRoot>_Complex_Solvated

--outputParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for generating output during energy minimization..

The supported parameter names along with their default values are shown below:

minimizationDataSteps, 100
 minimizationDataStdout, no [Possible values: yes or no]
 minimizationDataLog, yes [Possible values: yes or no]
 minimizationDataLogFile, auto [Default:
 <OutfilePrefix>_MinimizationOut.csv]
 minimizationDataOutType, auto [Possible values: A space delimited
 list of valid parameter names. Default: SystemEnergy
 RestraintEnergy MaxConstraintError.
 Other valid names: RestraintStrength]

 pdbOutFormat, PDB [Possible values: PDB or CIF]
 pdbOutKeepIDs, yes [Possible values: yes or no]

A brief description of parameters is provided below:

minimizationDataSteps: Frequency of writing data to stdout
 and log file.
 minimizationDataStdout: Write data to stdout.
 minimizationDataLog: Write data to log file.
 minimizationDataLogFile: Data log file name.
 minimizationDataOutType: Type of data to write to stdout
 and log file.

 pdbOutFormat: Format of output PDB files.
 pdbOutKeepIDs: Keep existing chain and residue IDs.

--overwrite

Overwrite existing files.

-p, --platform <text> [default: CPU]

Platform to use for running MD simulation. Possible values: CPU, CUDA, OpenCL, or Reference.

--platformParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs to configure platform for running energy minimization calculations..

The supported parameter names along with their default values for different platforms are shown below:

CPU:

 threads, 1 [Possible value: >= 0 or auto. The value of 'auto'

or zero implies the use of all available CPUs for threading.]

CUDA:

deviceIndex, auto [Possible values: 0, '0 1' etc.]
 deterministicForces, auto [Possible values: yes or no]
 precision, single [Possible values: single, double, or mix]
 tempDirectory, auto [Possible value: DirName]
 useBlockingSync, auto [Possible values: yes or no]
 useCpuPme, auto [Possible values: yes or no]

OpenCL:

deviceIndex, auto [Possible values: 0, '0 1' etc.]
 openCLPlatformIndex, auto [Possible value: Number]
 precision, single [Possible values: single, double, or mix]
 useCpuPme, auto [Possible values: yes or no]

A brief description of parameters is provided below:

CPU:

threads: Number of threads to use for simulation.

CUDA:

deviceIndex: Space delimited list of device indices to use for calculations.
 deterministicForces: Generate reproducible results at the cost of a small decrease in performance.
 precision: Number precision to use for calculations.
 tempDirectory: Directory name for storing temporary files.
 useBlockingSync: Control run-time synchronization between CPU and GPU.
 useCpuPme: Use CPU-based PME implementation.

OpenCL:

deviceIndex: Space delimited list of device indices to use for simulation.
 openCLPlatformIndex: Platform index to use for calculations.
 precision: Number precision to use for calculations.
 useCpuPme: Use CPU-based PME implementation.

--restraintAtoms <yes or no> [default: no]

Restraint atoms during energy minimization. The motion of specified atoms is restricted by adding a harmonic force that binds them to their starting positions. The atoms are not completely fixed unlike freezing of atoms. Their motion, however, is restricted and they are not able to move far away from their starting positions during energy minimization.

--restraintAtomsParams <Name,Value,...>

A comma delimited list of parameter name and value pairs for restraining atoms during energy minimization. You must specify these parameters for 'yes' value of '--restraintAtoms' option.

The supported parameter names along with their default values are shown below:

selection, none [Possible values: CAlphaProtein, Ions, Ligand, Protein, Residues, or Water]
 selectionSpec, auto [Possible values: A space delimited list of residue names]
 negate, no [Possible values: yes or no]

A brief description of parameters is provided below:

selection: Atom selection to restraint.
 selectionSpec: A space delimited list of residue names for selecting atoms to restraint. You must specify its value during 'Ligand' and 'Protein' value for 'selection'. The default values are automatically set for 'CAlphaProtein', 'Ions', 'Protein', and 'Water' values of 'selection' as shown below:

CAlphaProtein: List of standard protein residues from pdbfixer
 for selecting CAlpha atoms.
 Ions: Li Na K Rb Cs Cl Br F I
 Water: HOH
 Protein: List of standard protein residues from pdbfixer.

negate: Negate atom selection match to select atoms for freezing.

In addition, you may specify an explicit space delimited list of residue names using 'selectionSpec' for any 'selection'. The specified residue names are appended to the appropriate default values during the selection of atoms for restraining.

--restraintSpringConstant <number> [default: 2.5]

Restraint spring constant for applying external restraint force to restraint atoms relative to their initial positions during 'yes' value of '--restraintAtoms' option. Default units: kcal/mol/A**2. The default value, 2.5, corresponds to 1046.0 kJoules/mol/nm**2. The default value is automatically converted into units of kJoules/mol/nm**2 before its usage.

--simulationParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for simulation.

The supported parameter names along with their default values are shown below:

minimizationMaxSteps, auto [Possible values: >= 0. The value of zero implies until the minimization is converged.]
 minimizationTolerance, 0.24 [Units: kcal/mol/A. The default value 0.25, corresponds to OpenMM default of value of 10.04 kJoules/mol/nm. It is automatically converted into OpenMM default units before its usage.]

A brief description of parameters is provided below:

minimizationMaxSteps: Maximum number of minimization steps. The value of zero implies until the minimization is converged.
 minimizationTolerance: Energy convergence tolerance during minimization.

-s, --smallMolFile <SmallMolFile>

Small molecule input file name. The macromolecule and small molecule are merged for simulation and the complex is written out to a PDB file.

--smallMolID <text> [default: LIG]

Three letter small molecule residue ID. The small molecule ID corresponds to the residue name of the small molecule and is written out to a PDB file containing the complex.

--systemParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs to configure a system for energy minimization. No MD simulation is performed.

The supported parameter names along with their default values are shown below:

constraints, BondsInvolvingHydrogens [Possible values: None, WaterOnly, BondsInvolvingHydrogens, AllBonds, or AnglesInvolvingHydrogens]
 constraintErrorTolerance, 0.000001
 ewaldErrorTolerance, 0.0005

 nonbondedMethodPeriodic, PME [Possible values: NoCutoff, CutoffNonPeriodic, or PME]
 nonbondedMethodNonPeriodic, NoCutoff [Possible values: NoCutoff or CutoffNonPeriodic]
 nonbondedCutoff, 1.0 [Units: nm]

 hydrogenMassRepartitioning, yes [Possible values: yes or no]
 hydrogenMass, 1.5 [Units: amu]

 removeCMMotion, yes [Possible values: yes or no]
 rigidWater, auto [Possible values: yes or no. Default: 'No' for 'None' value of constraints; Otherwise, yes]

A brief description of parameters is provided below:

constraints: Type of system constraints to use for simulation. These constraints are different from freezing and restraining of any atoms in the system.

constraintErrorTolerance: Distance tolerance for constraints as a fraction of the constrained distance.

ewaldErrorTolerance: Ewald error tolerance for a periodic system.

nonbondedMethodPeriodic: Nonbonded method to use during the calculation of long range interactions for a periodic system.

nonbondedMethodNonPeriodic: Nonbonded method to use during the calculation of long range interactions for a non-periodic system.

nonbondedCutoff: Cutoff distance to use for long range interactions in both periodic non-periodic systems.

hydrogenMassRepartitioning: Use hydrogen mass repartitioning. It increases the mass of the hydrogen atoms attached to the heavy atoms and decreasing the mass of the bonded heavy atom to maintain constant system mass. This allows the use of larger integration step size (4 fs) during a simulation.

hydrogenMass: Hydrogen mass to use during repartitioning.

removeCMMotion: Remove all center of mass motion at every time step.

rigidWater: Keep water rigid during a simulation. This is determined automatically based on the value of 'constraints' parameter.

--waterBox <yes or no> [default: no]

Add water box.

--waterBoxParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for adding a water box.

The supported parameter names along with their default values are shown below:

model, tip3p [Possible values: tip3p, spce, tip4pew, tip5p or swm4ndp]

mode, Padding [Possible values: Size or Padding]

padding, 1.0

size, None [Possible value: xsize ysize zsize]

shape, cube [Possible values: cube, dodecahedron, or octahedron]

ionPositive, Na+ [Possible values: Li+, Na+, K+, Rb+, or Cs+]

ionNegative, Cl- [Possible values: Cl-, Br-, F-, or I-]

ionicStrength, 0.0

A brief description of parameters is provided below:

model: Water model to use for adding water box. The van der Waals radii and atomic charges are determined using the specified water forcefield. You must specify an appropriate water forcefield. No validation is performed.

mode: Specify the size of the waterbox explicitly or calculate it automatically for a macromolecule along with adding padding around the macromolecule.

padding: Padding around a macromolecule in nanometers for filling box with water. It must be specified during 'Padding' value of 'mode' parameter.

size: A space delimited triplet of values corresponding to water size in nanometers. It must be specified during 'Size' value of 'mode' parameter.

ionPositive: Type of positive ion to add during the addition of a water box.

ionNegative: Type of negative ion to add during the addition of a water box.

ionicStrength: Total concentration of both positive and negative ions to add excluding the ions added to neutralize the system during the addition of a water box.

-w, --workingdir <dir>

Location of working directory which defaults to the current directory.

EXAMPLES

To perform energy minimization for a macromolecule in a PDB file until the energy is converged, writing information to log file every 100 steps and generate a PDB file for the minimized system, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb
```

To run the first example for writing information to both stdout and log file every 100 steps and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb --outputParams  
"minimizationDataStdout, yes"
```

To run the first example for performing OpenMM simulation using multi- threading employing all available CPUs on your machine and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb  
--platformParams "threads,0"
```

To run the first example for performing OpenMM simulation using CUDA platform on your machine and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb -p CUDA
```

To run the second example for a macromolecule in a complex with a small molecule and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb -s Sample13Ligand.sdf  
--platformParams "threads,0"
```

To run the second example by adding a water box to the system and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb --waterBox yes  
--platformParams "threads,0"
```

To run the second example for a macromolecule in a complex with a small molecule by adding a water box to the system and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb -s Sample13Ligand.sdf  
--waterBox yes --platformParams "threads,0"
```

To run the second example by freezing CAlpha atoms in a macromolecule without using any system constraints to avoid any issues with the freezing of the same atoms and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb  
--freezeAtoms yes --freezeAtomsParams "selection,CAlphaProtein"  
--systemParams "constraints, None"  
--platformParams "threads,0"
```

```
% OpenMMPerformMinimization.py -i Sample13.pdb  
--freezeAtoms yes --freezeAtomsParams "selection,CAlphaProtein"  
--systemParams "constraints, None"  
--platformParams "threads,0" --waterBox yes
```

To run the second example by restraining CAlpha atoms in a macromolecule and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb  
--restraintAtomsParams "selection,CAlphaProtein"  
--platformParams "threads,0"
```

```
% OpenMMPerformMinimization.py -i Sample13.pdb  
--restraintAtomsParams "selection,CAlphaProtein"  
--platformParams "threads,0"  
--waterBox yes
```

To run the second example by specifying explicit values for various parameters and generate various output files, type:

```
% OpenMMPerformMinimization.py -i Sample13.pdb
-f " biopolymer,amber14-all.xml,smallMolecule, openff-2.2.1,
water,amber14/tip3pfb.xml" --waterBox yes
--outputParams "minimizationDataSteps, 100, minimizationDataStdout,
yes,minimizationDataLog,yes"
-p CPU --platformParams "threads,0"
--simulationParams "minimizationMaxSteps,500,
minimizationTolerance, 0.24"
--systemParams "constraints,BondsInvolvingHydrogens,
nonbondedMethodPeriodic,PME,nonbondedMethodNonPeriodic,NoCutoff,
hydrogenMassRepartitioning, yes"
```

AUTHOR

Manish Sud(msud@san.rr.com)

SEE ALSO

OpenMMPrepareMacromolecule.py, OpenMMPerformMDSimulation.py, OpenMMPerformSimulatedAnnealing.py

COPYRIGHT

Copyright (C) 2025 Manish Sud. All rights reserved.

The functionality available in this script is implemented using OpenMM, an open source molecular simulation package.

This file is part of MayaChemTools.

MayaChemTools is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 3 of the License, or (at your option) any later version.