NAME

Psi4PerformMinimization.py - Perform structure minimization

SYNOPSIS

Psi4PerformMinimization.py [--basisSet <text>] [--confGeneration <yes or no>] [--confParams <Name,Value,...>] [--energyOut <yes or no>] [--energyDataFieldLabel <text>] [--energyUnits <text>] [--infileParams <Name,Value,...>] [--maxIters <number>] [--methodName <text>] [--mp <yes or no>] [--mpLevel <Molecules or Conformers>] [--mpParams <Name, Value,...>] [--outfileParams <Name,Value,...>] [--overwrite] [--precision <number>] [--psi4OptionsParams <Name,Value,...>] [--psi4RunParams <Name,Value,...>] [--psi4ConstrainTorsions <yes or no>] [--psi4ConstrainTorsionsParams <Name,Value,...>] [--psi4DDXSolvation <yes or no>] [--psi4DDXSolvationParams <Name,Value,...>] [--quiet <yes or no>] [--reference <text>] [--recenterAndReorient <yes or no>] [--symmetrize <yes or no>] [--symmetrizeTolerance <number>] [-w <dir>] -i <infile> -o <outfile>

Psi4PerformMinimization.py --psi4DDXListSolvents

Psi4PerformMinimization.py -h | --help | -e | --examples

DESCRIPTION

Generate 3D structures for molecules using a combination of distance geometry and forcefield minimization followed by geometry optimization using a quantum chemistry method. A set of initial 3D structures are generated for a molecule employing distance geometry. The 3D structures in the conformation ensemble are sequentially minimized using forcefield and a quantum chemistry method. The conformation with lowest energy is selected to represent the final structure. An option is available to skip the generation of the conformation ensemble along with forcefield minimization and simply perform minimization on the initial 3D structure using a quantum chemistry method.

A Psi4 XYZ format geometry string is automatically generated for each molecule in input file. It contains atom symbols and 3D coordinates for each atom in a molecule. In addition, the formal charge and spin multiplicity are present in the the geometry string. These values are either retrieved from molecule properties named 'FormalCharge' and 'SpinMultiplicty' or dynamically calculated for a molecule.

The supported input file formats are: Mol (.mol), SD (.sdf, .sd), SMILES (.smi, .csv, .tsv, .txt)

The supported output file formats are: SD (.sdf, .sd)

OPTIONS

-b, --basisSet <text> [default: auto]

Basis set to use for energy minimization. Default: 6-31+G** for sulfur containing molecules; Otherwise, 6-31G** [Ref 150]. The specified value must be a valid Psi4 basis set. No validation is performed.

The following list shows a representative sample of basis sets available in Psi4:

```
STO-3G, 6-31G, 6-31+G, 6-31++G, 6-31G*, 6-31+G*,  6-31++G*,
6-31G**, 6-31+G**, 6-31++G**, 6-311G, 6-311+G, 6-311++G,
6-311G*, 6-311+G*, 6-311++G*, 6-311G**, 6-311+G**, 6-311++G**,
cc-pVDZ, cc-pCVDZ, aug-cc-pVDZ, cc-pVDZ-DK, cc-pCVDZ-DK, def2-SVP,
def2-SVPD, def2-TZVP, def2-TZVPD, def2-TZVPP, def2-TZVPPD
```

-c, --confGeneration <yes or no> [default: yes]

Generate an initial 3D conformation ensemble using distance geometry and forcefield minimization before final geometry optimization by a specified method name and basis set. Possible values: yes or no.

The 'No' value skips the generation of conformations employing distance geometry and forcefield minimization. The 3D structures in input file are minimized by a quantum method. It is not allowed for SMILES input file.

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

A comma delimited list of parameter name and value pairs for generating initial sets of 3D conformations for molecules. The 3D conformation ensemble is generated using distance geometry and forcefield functionality available in RDKit. The 3D structures in the conformation ensemble are subsequently minimized by a quantum chemistry method available in Psi4.

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

```
confMethod,ETKDGv2,
forceField,MMFF, forceFieldMMFFVariant,MMFF94,
enforceChirality,yes,embedRMSDCutoff,0.5,maxConfs,50,
maxIters,250,randomSeed,auto

confMethod,ETKDGv2   [ Possible values: SDG, KDG, ETDG,
    ETKDG , or ETKDGv2]
forceField, MMFF   [ Possible values: UFF or MMFF ]
forceFieldMMFFVariant,MMFF94   [ Possible values: MMFF94 or MMFF94s ]
enforceChirality,yes   [ Possible values: yes or no ]
embedRMSDCutoff,0.5   [ Possible values: number or None]
```

confMethod: Conformation generation methodology for generating initial 3D coordinates. Possible values: Standard Distance Geometry (SDG), Experimental Torsion-angle preference with Distance Geometry (ETDG), basic Knowledge-terms with Distance Geometry (KDG) and Experimental Torsion-angle preference along with basic Knowledge-terms and Distance Geometry (ETKDG or ETKDGv2) [Ref 129, 167] .

forceField: Forcefield method to use for energy minimization. Possible values: Universal Force Field (UFF) [Ref 81] or Merck Molecular Mechanics Force Field [Ref 83-87] .

enforceChirality: Enforce chirality for defined chiral centers during forcefield minimization.

maxConfs: Maximum number of conformations to generate for each molecule during the generation of an initial 3D conformation ensemble using conformation generation methodology. The conformations are minimized using the specified forcefield and a quantum chemistry method. The lowest energy conformation is written to the output file.

embedRMSDCutoff: RMSD cutoff for retaining initial set conformers embedded using distance geometry and before forcefield minimization. All embedded conformers are kept for 'None' value. Otherwise, only those conformers which are different from each other by the specified RMSD cutoff, 0.5 by default, are kept. The first embedded conformer is always retained.

maxIters: Maximum number of iterations to perform for each conformation during forcefield minimization.

randomSeed: Seed for the random number generator for reproducing initial 3D coordinates in a conformation ensemble. Default is to use a random seed.

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

Write out energy values.

--energyDataFieldLabel <text> [default: auto]

Energy data field label for writing energy values. Default: Psi4_Energy (<Units>).

--energyUnits <text> [default: kcal/mol]

Energy units. Possible values: Hartrees, kcal/mol, kJ/mol, or eV.

-e, --examples

Print examples.

-h, --help

Print this help message.

-i, --infile <infile>

Input file name.

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

A comma delimited list of parameter name and value pairs for reading molecules from files. The supported parameter names for different file formats, along with their default values, are shown below:

```
SD, MOL: removeHydrogens,no,sanitize,yes,strictParsing,yes
SMILES: smilesColumn,1,smilesNameColumn,2,smilesDelimiter,space,
    smilesTitleLine,auto,sanitize,yes
```

Possible values for smilesDelimiter: space, comma or tab.

--maxIters <number> [default: 50]

Maximum number of iterations to perform for each molecule or conformer during energy minimization by a quantum chemistry method.

-m, --methodName <text> [default: auto]

Method to use for energy minimization. Default: B3LYP [Ref 150]. The specified value must be a valid Psi4 method name. No validation is performed.

The following list shows a representative sample of methods available in Psi4:

```
B1LYP, B2PLYP, B2PLYP-D3BJ, B2PLYP-D3MBJ, B3LYP, B3LYP-D3BJ,
B3LYP-D3MBJ, CAM-B3LYP, CAM-B3LYP-D3BJ, HF, HF-D3BJ,  HF3c, M05,
M06, M06-2x, M06-HF, M06-L, MN12-L, MN15, MN15-D3BJ,PBE, PBE0,
PBEH3c, PW6B95, PW6B95-D3BJ, WB97, WB97X, WB97X-D, WB97X-D3BJ
```

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

Use multiprocessing.

By default, input data is retrieved in a lazy manner via mp.Pool.imap() function employing lazy RDKit data iterable. This allows processing of arbitrary large data sets without any additional requirements memory.

All input data may be optionally loaded into memory by mp.Pool.map() before starting worker processes in a process pool by setting the value of 'inputDataMode' to 'InMemory' in '--mpParams' option.

A word to the wise: The default 'chunkSize' value of 1 during 'Lazy' input data mode may adversely impact the performance. The '--mpParams' section provides additional information to tune the value of 'chunkSize'.

--mpLevel <Molecules or Conformers> [default: Molecules]

Perform multiprocessing at molecules or conformers level. Possible values: Molecules or Conformers. The 'Molecules' value starts a process pool at the molecules level. All conformers of a molecule are processed in a single process. The 'Conformers' value, however, starts a process pool at the conformers level. Each conformer of a molecule is processed in an individual process in the process pool. The default Psi4 'OutputFile' is set to 'quiet' using '--psi4RunParams' for 'Conformers' level. Otherwise, it may generate a large number of Psi4 output files.

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

A comma delimited list of parameter name and value pairs to configure multiprocessing.

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

```
chunkSize, auto
inputDataMode, Lazy   [ Possible values: InMemory or Lazy ]
numProcesses, auto   [ Default: mp.cpu_count() ]
```

These parameters are used by the following functions to configure and control the behavior of multiprocessing: mp.Pool(), mp.Pool.map(), and mp.Pool.imap().

The chunkSize determines chunks of input data passed to each worker process in a process pool by mp.Pool.map() and mp.Pool.imap() functions. The default value of chunkSize is dependent on the value of 'inputDataMode'.

The mp.Pool.map() function, invoked during 'InMemory' input data mode, automatically converts RDKit data iterable into a list, loads all data into memory, and calculates the default chunkSize using the following method as shown in its code:

```
chunkSize, extra = divmod(len(dataIterable), len(numProcesses) * 4)
if extra: chunkSize += 1
```

For example, the default chunkSize will be 7 for a pool of 4 worker processes and 100 data items.

The mp.Pool.imap() function, invoked during 'Lazy' input data mode, employs 'lazy' RDKit data iterable to retrieve data as needed, without loading all the data into memory. Consequently, the size of input data is not known a priori. It's not possible to estimate an optimal value for the chunkSize. The default chunkSize is set to 1.

The default value for the chunkSize during 'Lazy' data mode may adversely impact the performance due to the overhead associated with exchanging small chunks of data. It is generally a good idea to explicitly set chunkSize to a larger value during 'Lazy' input data mode, based on the size of your input data and number of processes in the process pool.

The mp.Pool.map() function waits for all worker processes to process all the data and return the results.

The mp.Pool.imap() function, however, returns the the results obtained from worker processes as soon as the results become available for specified chunks of data.

The order of data in the results returned by both mp.Pool.map() and mp.Pool.imap() functions always corresponds to the input data.

-o, --outfile <outfile>

Output file name.

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

A comma delimited list of parameter name and value pairs for writing molecules to files. The supported parameter names for different file formats, along with their default values, are shown below:

```
SD: kekulize,yes,forceV3000,no
```

--overwrite

Overwrite existing files.

--precision <number> [default: 6]

Floating point precision for writing energy values.

--psi4OptionsParams <Name,Value,...> [default: none]

A comma delimited list of Psi4 option name and value pairs for setting global and module options. The names are 'option_name' for global options and 'module_name__option_name' for options local to a module. The specified option names must be valid Psi4 names. No validation is performed.

The specified option name and value pairs are processed and passed to psi4.set_options() as a dictionary. The supported value types are float, integer, boolean, or string. The float value string is converted into a float. The valid values for a boolean string are yes, no, true, false, on, or off.

--psi4RunParams <Name,Value,...> [default: auto]

A comma delimited list of parameter name and value pairs for configuring Psi4 jobs.

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

```
MemoryInGB, 1
NumThreads, 1
OutputFile, auto   [ Possible  values: stdout, quiet, or FileName ]
ScratchDir, auto   [ Possivle values: DirName]
RemoveOutputFile, yes   [ Possible values: yes, no, true, or false]
```

These parameters control the runtime behavior of Psi4.

The default file name for 'OutputFile' is <InFileRoot>_Psi4.out. The PID is appended to output file name during multiprocessing as shown: <InFileRoot>_Psi4_<PIDNum>.out. The 'stdout' value for 'OutputType' sends Psi4 output to stdout. The 'quiet' or 'devnull' value suppresses all Psi4 output. The 'OutputFile' is set to 'quiet' for 'auto' value during 'Conformers' of '--mpLevel' option.

The default 'Yes' value of 'RemoveOutputFile' option forces the removal of any existing Psi4 before creating new files to append output from multiple Psi4 runs.

The option 'ScratchDir' is a directory path to the location of scratch files. The default value corresponds to Psi4 default. It may be used to override the deafult path.

--psi4ConstrainTorsions <yes or no> [default: no]

Constrain torsions around rotatable bonds during Psi4 minimization.

--psi4ConstrainTorsionsParams <Name,Value,...> [default: auto]

A space delimited list of parameter name and value pairs for constraining torsions around rotatable bonds during Psi4 minimization. The supported parameter names, along with their default values, are shown below:

```
ignoreHydrogens, yes,  [ Possible values: yes or no ]
rotBondsSMARTSMode, NonStrict,   [ Possible  values: NonStrict,
    SemiStrict, Strict, Specify ]
rotBondsSMARTSPattern, auto  [ Possible values: Valid SMARTS ]
```

ignoreHydrogens: Ignore torsions around rotatable bonds containing hydrogens during Psi4 minimization.

rotBondsSMARTSMode: SMARTS pattern to use for identifying rotatable bonds for constraining torsions during Psi4 minimization. Possible values: NonStrict, SemiStrict, Strict or Specify. The following SMARTS patterns are used to identify rotatable bonds for different modes:

```
NonStrict: [!$(*#*)&!D1]-&!@[!$(*#*)&!D1]

SemiStrict:
[!$(*#*)&!D1&!$(C(F)(F)F)&!$(C(Cl)(Cl)Cl)&!$(C(Br)(Br)Br)
&!$(C([CH3])([CH3])[CH3])]-!@[!$(*#*)&!D1&!$(C(F)(F)F)
&!$(C(Cl)(Cl)Cl)&!$(C(Br)(Br)Br)&!$(C([CH3])([CH3])[CH3])]

Strict:
[!$(*#*)&!D1&!$(C(F)(F)F)&!$(C(Cl)(Cl)Cl)&!$(C(Br)(Br)Br)
&!$(C([CH3])([CH3])[CH3])&!$([CD3](=[N,O,S])-!@[#7,O,S!D1])
&!$([#7,O,S!D1]-!@[CD3]=[N,O,S])&!$([CD3](=[N+])-!@[#7!D1])
&!$([#7!D1]-!@[CD3]=[N+])]-!@[!$(*#*)&!D1&!$(C(F)(F)F)
&!$(C(Cl)(Cl)Cl)&!$(C(Br)(Br)Br)&!$(C([CH3])([CH3])[CH3])]
```

The 'NonStrict' and 'Strict' SMARTS patterns are available in RDKit. The 'NonStrict' SMARTS pattern corresponds to original Daylight SMARTS specification for rotatable bonds. The 'SemiStrict' SMARTS pattern is derived from 'Strict' SMARTS patterns for its usage in this script.

You may use any arbitrary SMARTS pattern to identify rotatable bonds by choosing 'Specify' value for 'rotBondsSMARTSMode' parameter and providing its value via '--rotBondsSMARTSPattern'.

rotBondsSMARTSPattern: SMARTS pattern for identifying rotatable bonds for constraining torsions during Psi4 minimization. This option is only valid for 'Specify' value of 'rotBondsSMARTSMode' parameter.

--psi4DDXSolvation <yes or no> [default: no]

Perform minimization in solution using domain-decomposition-based continuum solvation models [Ref 160-161]. The script relies on Psi4 interface to the DDX module to perform these calculations. The DDX library provides a linear-scaling implementation of standard continuum solvation models using a domain-decomposition ansatz. Two solvation models are supported: COnductor-like Screening MOdel (COSMO) [Ref 162-163] and Polarizable Continuum Model (PCM) [Ref 164-165].

The solvation energy is included in the value of the total energy written to the output SD file. In addition, the value of solvation energy is written to the output file under its own data field label.

Psi4 relies on Python module PYDDX to calculate solvation energy. It must be present in your environment.

--psi4DDXSolvationParams <Name,Value,...> [default: auto]

A space delimited list of parameter name and value pairs for calculating solvation energy using the DDX module. The supported parameter names, along with their default values, are shown below:

```
solvationModel PCM   [ Possible  values: COSMO or PCM]
solvent water   [ Possible  values: A valid solvent name]
solventEpsilon None
radiiSet UFF   [ Possible  values: Bondi or UFF]
radiiScaling auto   [ Default: 1.2 for Bondi; 1.1 for UFF]

solvationModel: Solvation model for calculating solvation energy.
The corresponding Psi4 option is DDX_MODEL.

solvent: Solvent to use. The corresponding Ps4 option is
DDX_SOLVENT

solventEpsilon: Dielectric constant of the solvent. The
corresponding Psi4 option is DDX_SOLVENT_EPSILON.

radiiSet: Radius set for cavity spheres. The corresponding Psi4
option is DDX_RADII_SET.

radiiScaling: Scaling factor for cavity spheres. The default value
depends on radiiSet: 1.2 for Bondi; 1.1 for UFF. The corresponding
Psi4 option is DDX_RADII_SCALING.
```

These parameter names are automatically mapped to appropriate DDX keywords.

You may specify the solvent either by directly providing a dielectric constant using 'solventEpsilon' parameter or a solvent name corresponding to 'solvent' parameter. The solvent name must be a valid named supported by the DDX module. For example: water, ethanol, dimethylsulfoxide, cis-1,2-dimethylcyclohexane , etc. The 'solvent' parameter is ignored for non-zero value of 'solvent' option.

The DDX module contains a variety of addtional options to confgure the calculation of solvation energy. For example: DDX_MAXITER, DDX_RADII_SCALING, etc. You may use '--psi4OptionsParams' to modify values for additional DDX options.

--psi4DDXListSolvents

List solvent names, along with dielectric values, supported by the DDX module for the calculation of solvent energy without performing any calculation.

-q, --quiet <yes or no> [default: no]

Use quiet mode. The warning and information messages will not be printed.

-r, --reference <text> [default: auto]

Reference wave function to use for energy calculation. Default: RHF or UHF. The default values are Restricted Hartree-Fock (RHF) for closed-shell molecules with all electrons paired and Unrestricted Hartree-Fock (UHF) for open-shell molecules with unpaired electrons.

The specified value must be a valid Psi4 reference wave function. No validation is performed. For example: ROHF, CUHF, RKS, etc.

The spin multiplicity determines the default value of reference wave function for input molecules. It is calculated from number of free radical electrons using Hund's rule of maximum multiplicity defined as 2S + 1 where S is the total electron spin. The total spin is 1/2 the number of free radical electrons in a molecule. The value of 'SpinMultiplicity' molecule property takes precedence over the calculated value of spin multiplicity.

--recenterAndReorient <yes or no> [default: auto]

Recenter and reorient a molecule during creation of a Psi4 molecule from a geometry string. Default: 'No' during 'No' value of '--confGeneration'; Otherwise, 'Yes'.

The 'No' values allows the minimization of a molecule in its initial 3D coordinate space in input file or conformer ensemble.

--symmetrize <yes or no> [default: auto]

Symmetrize molecules before energy minimization. Default: 'Yes' during 'Yes' value of '--recenterAndReorient'; Otherwise, 'No'. The psi4 function, psi4mol.symmetrize(SymmetrizeTolerance), is called to symmetrize the molecule before calling psi4.optimize().

The 'No' value of '--symmetrize' during 'Yes' value of '--recenterAndReorient' may cause psi4.optimize() to fail with a 'Point group changed...' error message.

--symmetrizeTolerance <number> [default: 0.01]

Symmetry tolerance for '--symmetrize'.

-w, --workingdir <dir>

Location of working directory which defaults to the current directory.

EXAMPLES

To generate an initial conformer ensemble of up to 50 conformations using a combination of ETKDGv2 distance geometry methodology, applying RMSD cutoff of 0.5 and MMFF forcefield minimization, followed by energy minimization using B3LYP/6-31G** and B3LYP/6-31+G** for non-sulfur and sulfur containing molecules in a SMILES file and write out a SD file containing minimum energy structure corresponding to each molecule, type:

```
% Psi4PerformMinimization.py -i Psi4Sample.smi -o Psi4SampleOut.sdf
```

To run the first example in a quiet mode and write out a SD file, type:

```
% Psi4PerformMinimization.py -q yes -i Psi4Sample.smi -o
  Psi4SampleOut.sdf
```

To run the first example along with performing energy mimimization in solution using default DDX solvation parameters for water and write out a SD file, type:

```
% Psi4PerformMinimization.py -i Psi4Sample.smi -o Psi4SampleOut.sdf
  --psi4DDXSolvation yes
```

To run the first example in multiprocessing mode at molecules level on all available CPUs without loading all data into memory and write out a SD file, type:

```
% Psi4PerformMinimization.py --mp yes -i Psi4Sample.smi -o
  Psi4SampleOut.sdf
```

To run the first example in multiprocessing mode at conformers level on all available CPUs without loading all data into memory and write out a SD file, type:

```
% Psi4PerformMinimization.py --mp yes --mpLevel Conformers
  -i Psi4Sample.smi -o Psi4SampleOut.sdf
```

To run the first example in multiprocessing mode at molecules level on all available CPUs by loading all data into memory and write out a SD file, type:

```
% Psi4PerformMinimization.py  --mp yes --mpParams "inputDataMode,
  InMemory" -i Psi4Sample.smi -o Psi4SampleOut.sdf
```

To run the first example in multiprocessing mode at molecules level on specific number of CPUs and chunk size without loading all data into memory and write out a SD file, type:

```
% Psi4PerformMinimization.py  --mp yes --mpParams "inputDataMode,Lazy,
  numProcesses,4,chunkSize,8" -i Psi4Sample.smi -o Psi4SampleOut.sdf
```

To generate an initial conformer ensemble of up to 20 conformations using a combination of ETKDGv2 distance geometry methodology and MMFF94s forcefield minimization followed by energy minimization for a maxium of 20 iterations using B3LYP/6-31+G** molecules in a SMILES file and write out a SD file containing minimum energy structure along with energy in specific units, type:

```
% Psi4PerformMinimization.py --confGeneration yes --confParams
  "confMethod,ETKDGv2,forceField,MMFF, forceFieldMMFFVariant,MMFF94s,
  maxConfs,20,embedRMSDCutoff,0.5" --energyUnits "kJ/mol" -m B3LYP
  -b "6-31+G**" --maxIters 20 -i Psi4Sample.smi -o Psi4SampleOut.sdf
```

To minimize molecules in a 3D files using B3LYP/6-31G** and B3LYP/6-31+G** for non-sulfur and sulfur containing molecules for a maximum of 25 iterations without generating any conformations and write out a SD file containing minimum energy structures corresponding to each molecule, type:

```
% Psi4PerformMinimization.py --confGeneration no --maxIters 25
  -i Psi4Sample3D.sdf -o Psi4Sample3DOut.sdf
```

To run the first example for molecules in a CSV SMILES file, SMILES strings in column 1, name column 2, and write out a SD file, type:

```
% Psi4PerformMinimization.py --infileParams "smilesDelimiter,comma,
  smilesTitleLine,yes,smilesColumn,1,smilesNameColumn,2"
  -i Psi4Sample.csv -o Psi4SampleOut.sdf
```

AUTHOR

Manish Sud(msud@san.rr.com)

CONTRIBUTOR

Francois Berenger

SEE ALSO

Psi4CalculateEnergy.py, Psi4CalculatePartialCharges.py, Psi4GenerateConformers.py

COPYRIGHT

Copyright (C) 2026 Manish Sud. All rights reserved.

The functionality available in this script is implemented using Psi4, an open source quantum chemistry software package, and RDKit, an open source toolkit for cheminformatics developed by Greg Landrum.

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.