

NAME

Psi4CalculateInteractionEnergy.py - Calculate interaction energy

SYNOPSIS

```
Psi4CalculateInteractionEnergy.py [--basisSet <text>] [--bsseType <CP, noCP, VMFC, or None>] [
--energyDataFieldLabel <text>] [--energySAPTDDataFieldLabels <Type,Label,...>] [--energyUnits
<text>] [--infileParams <Name,Value,...>] [--methodName <text>] [--mp <yes or no>] [--mpParams
<Name, Value,...>] [--outfileParams <Name,Value,...>] [--overwrite] [--precision <number>] [
--psi4OptionsParams <Name,Value,...>] [--psi4RunParams <Name,Value,...>] [--quiet <yes or no>] [
--reference <text>] [-w <dir>] -i <infile> -o <outfile>
```

Psi4CalculateInteractionEnergy.py -h | --help | -e | --examples

DESCRIPTION

Calculate interaction energy for molecules using a specified method name and basis set. The molecules must contain exactly two fragments or disconnected components for Symmetry Adapted Perturbation Theory (SAPT) [Ref 154-155] and Spin-Network-Scaled MP2 (SNS-MP2) [Ref 156] calculations and more than one fragment for all other calculations. An arbitrary number of fragments may be present in a molecule for Basis Set Superposition Energy (BSSE) correction calculations.

The interaction energy is calculated at SAPTO / jun-cc-pVDZ level of theory by default. The SAPTO calculations are relatively fast but less accurate. You may want to consider calculating interaction energy at WB97X-D3 / aug-cc-pVTZ, B3LYP-D3 / aug-cc-pVTZ, or higher level of theory [Ref 157] to improve the accuracy of your results. The WB97X-D3 and B3LYP-D3 calculations rely on the presence of DFTD3 and gCP Psi4 plugins in your environment.

The molecules must have 3D coordinates in input file. The molecular geometry is not optimized before the calculation. In addition, hydrogens must be present for all molecules in input file. The 3D coordinates are not modified during the calculation.

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 'SpinMultiplicity' or dynamically calculated for a molecule. A double dash separates each fragment or component in a molecule. The same formal charge and multiplicity values are assigned to each fragment in a molecule.

The supported input file formats are: Mol (.mol), SD (.sdf, .sd)

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

OPTIONS

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

Basis set to use for interaction energy calculation. Default: jun-cc-pVDZ for SAPT calculations; None for SNS-MP2 calculations to use its default basis set; otherwise, it must be explicitly specified using this option. The specified value must be a valid Psi4 basis set. No validation is performed. You may set an empty string as a value for the basis set.

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
```

--bsseType <CP, noCP, VMFC, or None> [default: auto]

Type of Basis Set Superposition Energy (BSSE) correction to apply during the calculation of interaction energy. Possible values:

```
CP: Counterpoise corrected interaction energy
noCP: Supramolecular interaction energy without any CP correction
VMFC: Valiron-Mayer Function Counterpoise correction
None: The Psi4 option 'bsse_type' is not passed to the energy
```

function during the calculation of interaction energy

Default values:

None: SAPT and SNS-MP2 calculations. An explicit bsse_type option is not valid for these calculations.
 HF3c: noCP to use built-in correction
 CP: All other calculations

--energyDataFieldLabel <text> [default: auto]

Energy data field label for writing interaction energy values. Default: Psi4_SAPT_Interaction_Energy (<Units>) for SAPT calculation; Psi4_SNS-MP2_Interaction_Energy (<Units>) for SNS-MP2 calculation; otherwise, Psi4_Interaction_Energy (<Units>)

--energySAPTDATAFieldLabels <Type,Label,...> [default: auto]

A comma delimited interaction energy type and data field label value pairs for writing individual components of total SAPT interaction energy.

The supported SAPT energy types along with their default data field label values are shown below:

ElectrostaticEnergy, Psi4_SAPT_Electrostatic_Energy (<Units>),
 ExchangeEnergy, Psi4_SAPT_Exchange_Energy (<Units>),
 InductionEnergy, Psi4_SAPT_Induction_Energy (<Units>),
 DispersionEnergy, Psi4_SAPT_Dispersion_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

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

Method to use for interaction energy calculation. Default: SAPT0. 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:

SAPT0, SAPT2, SAPT2+, SAPT2+(CCD), SAPT2+DMP2, SAPT2+(CCD)DMP2
 SAPT2+(3), SAPT2+(3)(CCD), SAPT2+DMP2, SAPT2+(CCD)DMP2,
 SAPT2+3, SAPT2+3(CCD), SAPT2+(3)DMP2, SAPT2+3(CCD)DMP2, SNS-MP2,
 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'.

--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 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.

-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 interaction 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.

The 'SpinMultiplicity' molecule property may contain multiples values corresponding to the number of fragments in a molecule. This property must not be set for automatic determination of the reference wave functionn.

-w, --workingdir <dir>

Location of working directory which defaults to the current directory.

EXAMPLES

To calculate interaction energy using SAPTO/aug-cc-pVDZ for molecules in a SD file, use RHF and UHF for closed-shell and open-shell molecules, and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -i Psi4SampleDimers3D.sdf
-o Psi4SampleDimers3DOut.sdf
```

To run the first example for freezing core electrons and setting SCF type to DF and write out a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py --psi4OptionsParams "scf_type, df,
freeze_core, true" -i Psi4SampleDimers3D.sdf -o
Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy using SNS2-MP methodology for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m "sns-mp2"
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy at WB97X-D3/aug-cc-pVTZ level of theory, along with explicit Psi4 run time parameters, for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m WB97X-D3 -b aug-cc-pVTZ
--bsseType CP --psi4RunParams "NumThreads,4,MemoryInGB,6"
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy at B3LYP-D3/aug-cc-pVTZ level of theory using default Psi4 run time parameters for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m B3LYP-D3 -b aug-cc-pVTZ
--bsseType CP -i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy at B3LYP-D3/aug-cc-pVTZ level of theory, along with specifying grid resolution using Psi4 options and explicit Psi4 run time parameters, for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m B3LYP-D3 -b aug-cc-pVTZ
--bsseType CP --psi4OptionsParams "dft_spherical_points, 302,
dft_radial_points, 75" --psi4RunParams "NumThreads,4,MemoryInGB,6"
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy at HF3c level of theory using built-in basis set for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m HF3c -b "" --bsseType noCP
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To calculate interaction energy at CCSD(T)/aug-cc-pVDZ level of theory using default Psi4 run time parameters for molecules in a SD containing 3D structures and write a new SD file, type:

```
% Psi4CalculateInteractionEnergy.py -m "ccsd(t)" -b "aug-cc-pvdz"
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

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

```
% Psi4CalculateInteractionEnergy.py --mp yes -i Psi4SampleDimers3D.sdf
-o Psi4SampleDimers3DOut.sdf
```

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

```
% Psi4CalculateInteractionEnergy.py --mp yes --mpParams "inputDataMode,
InMemory" -i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

To run the first example in multiprocessing mode on all available CPUs without loading all data into memory along with multiple threads for each Psi4 run and write out a SD file, type:

```
% Psi4CalculateInteractionEnergy.py --mp yes --psi4RunParams "NumThreads,2"
-i Psi4SampleDimers3D.sdf -o Psi4SampleDimers3DOut.sdf
```

AUTHOR

Manish Sud(msud@san.rr.com)

SEE ALSO

Psi4CalculateEnergy.py, Psi4CalculatePartialCharges.py, Psi4PerformMinimization.py, Psi4GenerateConformers.py

COPYRIGHT

Copyright (C) 2025 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.