

---

# **ChemOpt Documentation**

*Release 0.1.0*

**Oskar Weser**

**Mar 15, 2018**



<b>1</b>	<b>Installation</b>	<b>1</b>
<b>2</b>	<b>Documentation</b>	<b>3</b>
2.1	Optimiser . . . . .	3
2.1.1	In non-redundant internal coordinates (Zmatrix) . . . . .	3
2.1.1.1	optimise . . . . .	3
2.2	Interfaces for electronic structure calculation . . . . .	4
2.2.1	Generic Interface . . . . .	4
2.2.1.1	calculate . . . . .	5
2.2.2	Molcas Interface . . . . .	5
2.2.2.1	calculate . . . . .	5
2.2.2.2	generate_input_file . . . . .	6
2.2.2.3	parse_output . . . . .	7
2.2.3	Molpro Interface . . . . .	7
2.2.3.1	calculate . . . . .	7
2.2.3.2	generate_input_file . . . . .	8
2.2.3.3	parse_output . . . . .	8



# CHAPTER 1

---

## Installation

---

For the installation checkout the [git repository](#) and execute the following command:

```
pip install .
```



## 2.1 Optimiser

### 2.1.1 In non-redundant internal coordinates (Zmatrix)

---

<code>optimise(zmolecule, hamiltonian, basis[, ...])</code>	Optimize a molecule.
---	----------------------

---

#### 2.1.1.1 optimise

`chemopt.zmat_optimisation.optimise` (*zmolecule*, *hamiltonian*, *basis*, *symbols=None*, *md\_out=None*, *el\_calc\_dir=None*, *molden\_out=None*, *etol=1e-06*, *gtol=0.0006*, *max\_iter=100*, *backend='molcas'*, *charge=0*, *title=""*, *multiplicity=1*, *num\_procs=None*, *mem\_per\_proc=None*, *start\_orb=None*, *coord\_fmt='.4f'*, *\*\*kwargs*)

Optimize a molecule.

#### Parameters

- **zmolecule** (*chemcoord.Zmat*) –
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.
- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **symbols** (*list*) – A list of tuples. Each tuple consists of a sympy symbolic expression and a starting value. An example is: [(*r*, 1.1), (*alpha*, 120)]. Has exactly the same format as the multi-parameter substitution in sympy.
- **el\_calc\_dir** (*str*) – Specify the input filename for electronic calculations. If it is None, the filename of the calling python script is used (With the suffix `.inp` instead of `.py`) and the files for the electronic calculations will reside in their own directory.

- **md\_out** (*str*) – Specify the output filename for chemopt output files. If it is None, the filename of the calling python script is used (With the suffix `.md` instead of `.py`). The output will be `os.path.splitext(inputfile)[0] + '.md'`.
- **molden\_out** (*str*) – Specify the output filename for the molden file from a geometry optimisation. If it is None, the filename of the calling python script is used (With the suffix `.molden` instead of `.py`). The output will be `os.path.splitext(inputfile)[0] + '.molden'`.
- **backend** (*str*) – Specify which QM program suite should be used. Allowed values are {'molpro', 'molcas'}, the default is 'molcas'.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **title** (*str*) – The title to be printed in input and output.
- **multiplicity** (*int*) – The spin multiplicity. The default is 1.
- **etol** (*float*) – Convergence criterium for the energy.
- **gtol** (*float*) – Convergence criterium for the gradient.
- **max\_iter** (*int*) – Maximum number of iterations. The default is '100'.
- **num\_procs** (*int*) – The number of processes to spawn.
- **mem\_per\_proc** (*str*) – Memory per process. This is a string with a number and a unit like '800 MB'. SI and binary prefixes are supported. Uses the `datasize` library for parsing.
- **start\_orb** (*str*) – Path to an orbital file, if starting orbitals should be used.
- **coord\_fmt** (*str*) – A string as float formatter for the coordinates in the output file of chemopt. The default is '.4f'

### Returns

A list of dictionaries. The last one is the optimised structure. The keys of each dictionary depend on the used optimisation. In any case each dictionary has at least two keys:

- 'energy': The energy in Hartree.
- 'structure': The Zmatrix.

If `symbols` was None a generic optimisation was performed and the following keys are available:

- 'grad\_energy': The energy gradient ('grad\_energy')

in internal coordinates. The units are Hartree / Angstrom for bonds and Hartree / radians for angles and dihedrals.

If `symbols` was not None an optimisation with reduced degrees of freedom was performed and the following keys are available:

- 'symbols': A list of tuples containing the symbol and its value.

Return type `list`

## 2.2 Interfaces for electronic structure calculation

### 2.2.1 Generic Interface



---

<code>calculate(molecule, hamiltonian, basis[, ...])</code>	Calculate the energy of a molecule.
---	-------------------------------------

---

### 2.2.1.1 calculate

`chemopt.interface.generic.calculate` (*molecule*, *hamiltonian*, *basis*, *el\_calc\_input=None*, *backend=None*, *charge=0*, *forces=False*, *title=""*, *multiplicity=1*, *\*\*kwargs*)

Calculate the energy of a molecule.

#### Parameters

- **molecule** (*chemcoord.Cartesian* or *chemcoord.Zmat* or *str*) – If it is a string, it has to be a valid xyz-file.
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.
- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **el\_calc\_input** (*str*) – Specify the input filename for electronic calculations. If it is None, the filename of the calling python script is used (With the suffix `.inp` instead of `.py`). The output will be `os.path.splitext(inputfile)[0] + '.inp'`.
- **backend** (*str*) – Specify which QM program suite should be used. Allowed values are {'molpro', 'molcas'}, the default is 'molcas'.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **forces** (*bool*) – Specify if energy gradients should be calculated. The default is False.
- **title** (*str*) – The title to be printed in input and output.
- **multiplicity** (*int*) – The spin multiplicity. The default is 1.

**Returns** A dictionary with at least the keys 'structure' and 'energy' which contains the energy in Hartree. If forces were calculated, the key 'gradient' contains the gradient in Hartree / Angstrom.

**Return type** `dict`

## 2.2.2 Molcas Interface

---

<code>calculate(molecule, hamiltonian, basis[, ...])</code>	Calculate the energy of a molecule using Molcas.
<code>generate_input_file(molecule, hamiltonian, ...)</code>	Generate a molcas input file.
<code>parse_output(output_path)</code>	Parse a molcas output file.

---

### 2.2.2.1 calculate

`chemopt.interface.molcas.calculate` (*molecule*, *hamiltonian*, *basis*, *molcas\_exe=None*, *el\_calc\_input=None*, *sym\_group=None*, *charge=0*, *forces=False*, *title=""*, *multiplicity=1*, *start\_orb=None*, *num\_procs=None*, *mem\_per\_proc=None*)

Calculate the energy of a molecule using Molcas.

#### Parameters

- **el\_calc\_input** (*str*) – Specify the input filename for electronic calculations. If it is

None, the filename of the calling python script is used (With the suffix `.inp` instead of `.py`). The output will be `os.path.splitext(inputfile)[0] + '.inp'`.

- **molecule** (*chemcoord.Cartesian* or *chemcoord.Zmat* or *str*) – If it is a string, it has to be a valid xyz-file.
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.

But 'CCSD' and 'CCSD(T)' are not yet implemented.

- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **molcas\_exe** (*str*) – Specify the command to invoke molcas. The default is 'molcas'.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **forces** (*bool*) – Specify if energy gradients should be calculated. The default is False.
- **title** (*str*) – The title to be printed in input and output.
- **multiplicity** (*int*) – The spin multiplicity. The default is 1.
- **start\_orb** (*str*) – Path to an orbital file, if starting orbitals should be used.
- **num\_procs** (*int*) – The number of processes to spawn.
- **mem\_per\_proc** (*str*) – Memory per process. This is a string with a number and a unit like '800 MB'. SI and binary prefixes are supported. Uses the [datasize library](#) for parsing.

**Returns** A dictionary with at least the keys 'structure' and 'energy' which contains the energy in Hartree. If forces were calculated, the key 'gradient' contains the gradient in Hartree / Angstrom.

**Return type** `dict`

### 2.2.2.2 generate\_input\_file

```
chemopt.interface.molcas.generate_input_file(molecule,          hamiltonian,          basis,
                                             el_calc_input, charge=0, forces=False, title="",
                                             start_orb=None, sym_group=None,
                                             multiplicity=1)
```

Generate a molcas input file.

#### Parameters

- **molecule** (*chemcoord.Cartesian* or *chemcoord.Zmat* or *str*) – If it is a string, it has to be a valid xyz-file.
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.
- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **forces** (*bool*) – Specify if energy gradients should be calculated. The default is False.
- **title** (*str*) – The title to be printed in input and output.
- **multiplicity** (*int*) – The spin multiplicity. The default is 1.

- **wfn\_symmetry** (*int*) – The symmetry of the wavefunction specified with the molpro notation.

**Returns** molcas input.

**Return type** *str*

### 2.2.2.3 parse\_output

`chemopt.interface.molcas.parse_output(output_path)`

Parse a molcas output file.

**Parameters** **output\_path** (*str*) –

**Returns** A dictionary with at least the keys 'structure' and 'energy' which contains the energy in Hartree. If forces were calculated, the key 'gradient' contains the gradient in Hartree / Angstrom.

**Return type** *dict*

## 2.2.3 Molpro Interface

<code>calculate(molecule, hamiltonian, basis[, ...])</code>	Calculate the energy of a molecule using Molpro.
<code>generate_input_file(molecule, hamiltonian, basis)</code>	Generate a molpro input file.
<code>parse_output(output_path)</code>	Parse a molpro output file.

### 2.2.3.1 calculate

`chemopt.interface.molpro.calculate(molecule, hamiltonian, basis, molpro_exe=None, el_calc_input=None, charge=0, forces=False, title="", multiplicity=1, wfn_symmetry=1, num_procs=None, mem_per_proc=None)`

Calculate the energy of a molecule using Molpro.

#### Parameters

- **el\_calc\_input** (*str*) – Specify the input filename for electronic calculations. If it is None, the filename of the calling python script is used (With the suffix `.inp` instead of `.py`). The output will be `os.path.splitext(inputfile)[0] + '.inp'`.
- **molecule** (*chemcoord.Cartesian* or *chemcoord.Zmat* or *str*) – If it is a string, it has to be a valid xyz-file.
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.  
But 'RASSCF' and 'CASPT2' not yet implemented.
- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **molpro\_exe** (*str*) – Specify the command to invoke molpro. The default is 'molpro'.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **forces** (*bool*) – Specify if energy gradients should be calculated. The default is False.
- **title** (*str*) – The title to be printed in input and output.

- **multiplicity** (*int*) – The spin multiplicity. The default is 1.
- **wfn\_symmetry** (*int*) – The symmetry of the wavefunction specified with the molpro notation.
- **num\_procs** (*int*) – The number of processes to spawn.
- **mem\_per\_proc** (*str*) – Memory per process. This is a string with a number and a unit like ‘800 MB’. SI and binary prefixes are supported. Uses the [datasize library](#) for parsing.

**Returns** A dictionary with at least the keys 'structure' and 'energy' which contains the energy in Hartree. If forces were calculated, the key 'gradient' contains the gradient in Hartree / Angstrom.

**Return type** `dict`

### 2.2.3.2 generate\_input\_file

`chemopt.interface.molpro.generate_input_file` (*molecule, hamiltonian, basis, charge=0, forces=False, title="", multiplicity=1, wfn\_symmetry=1, mem\_per\_proc=None*)

Generate a molpro input file.

#### Parameters

- **molecule** (*chemcoord.Cartesian or chemcoord.Zmat or str*) – If it is a string, it has to be a valid xyz-file.
- **hamiltonian** (*str*) – The hamiltonian to use for calculating the electronic energy. The allowed values are {'SCF', 'RASSCF', 'CCSD', 'CASPT2', 'MP2', 'B3LYP', 'CCSD(T')}.
- **basis** (*str*) – The basis set to use for calculating the electronic energy.
- **charge** (*int*) – The overall charge of the molecule. The default is 0.
- **forces** (*bool*) – Specify if energy gradients should be calculated. The default is False.
- **title** (*str*) – The title to be printed in input and output.
- **multiplicity** (*int*) – The spin multiplicity. The default is 1.
- **wfn\_symmetry** (*int*) – The symmetry of the wavefunction specified with the molpro notation.
- **mem\_per\_proc** (*str*) – Memory per process. This is a string with a number and a unit like ‘800 MB’. SI and binary prefixes are supported. Uses the [datasize library](#) for parsing.

**Returns** Molpro input.

**Return type** `str`

### 2.2.3.3 parse\_output

`chemopt.interface.molpro.parse_output` (*output\_path*)

Parse a molpro output file.

**Parameters** `output_path` (*str*) –

**Returns** A dictionary with at least the keys 'structure' and 'energy' which contains the energy in Hartree. If forces were calculated, the key 'gradient' contains the gradient in Hartree / Angstrom.

**Return type** dict



## C

calculate() (in module chemopt.interface.generic), 5

calculate() (in module chemopt.interface.molcas), 5

calculate() (in module chemopt.interface.molpro), 7

## G

generate\_input\_file() (in module  
chemopt.interface.molcas), 6

generate\_input\_file() (in module  
chemopt.interface.molpro), 8

## O

optimise() (in module chemopt.zmat\_optimisation), 3

## P

parse\_output() (in module chemopt.interface.molcas), 7

parse\_output() (in module chemopt.interface.molpro), 8