MacroQC: An Electronic Structure Theory Software for Large-Scale Applications

User’s Manual

Uğur Bozkaya, Betül Ermiş, Yavuz Alagöz, Ash Ünal, and Ali Kaan Uyar

Department of Chemistry, Hacettepe University, Ankara 06800, Turkey

MacroQC Version: 1.0.2
Created on: October 22, 2021
Git Version: cbf52a2
Report bugs to: macroqcprog@gmail.com
# Contents

1 Overview ........................................ 5

1.1 About ........................................ 5

1.2 Capabilities .................................. 7

2 Citing MacroQC .................................... 8

2.1 Overall MacroQC Package ......................... 8

2.2 Orbital-Optimized Post-Hartree-Fock Methods ........... 8

2.2.1 Orbital-Optimized Second-Order Perturbation Theory (OMP2) .... 8

2.2.2 Orbital-Optimized Third-Order Perturbation Theory (OMP3) and Orbital-Optimized MP2.5 (OMP2.5) ............. 9

2.2.3 Orbital-Optimized Linearized Coupled-Cluster Doubles Method (OL-CCD) ........................................ 9

2.2.4 Orbital-Optimized Coupled-Cluster Doubles Method (OCCD) .... 10

2.2.5 Orbital-Optimized Coupled-Cluster Doubles Method with Perturbative Triples [OCCD(T) and OCCD(T)$_A$] .............. 10

2.3 Perturbation Theory (PT) ........................ 10

2.4 Coupled-Cluster (CC) ........................... 10

2.5 Second-Order Quasidegenerate Perturbation Theory (QDPT2) ............ 11

2.6 Extended Koopmans’ Theorem (EKT) ................. 11

2.7 Molint Framework .............................. 12

2.8 Dipole Moments and One-Electron Properties ............... 12

2.9 Anharmonic Force Field and IR Spectra ................. 12

2.10 Linear-Scaling Systematic Molecular Fragmentation ............ 12

2.11 Equation-of-Motion Coupled-Cluster ............................ 12

3 Download and Installation .......................... 13

3.1 Install with Conda (Linux and Mac OS X) ............... 13

3.2 Install with Self-Extracting Package (Linux and Mac OS X) .......... 13

3.3 Manual Installation from Zip Archive (Linux and Mac OS X) .......... 13

4 Testing MacroQC .................................... 14

5 Running MacroQC .................................... 14

5.1 Setting Environmental Variables ..................... 14
5.2 Running MacroQC in Serial Mode ........................................... 15
5.3 Running MacroQC in Parallel Mode Using OpenMP .................... 15

6 Basic Input Format ................................................................. 15
6.1 Input Examples .................................................................. 17
6.2 Ghost Atoms .................................................................... 17
6.3 External Charges ............................................................... 18
6.4 Multiple Jobs .................................................................... 18
6.5 Global Options .................................................................. 20
6.6 Default Values for the Frozen Core Approximation .................. 24

7 Basis Sets .............................................................................. 25
7.1 Overview ........................................................................ 25
7.2 Built-In Basis Sets ............................................................. 25
7.3 Assigning Basis Sets for Individual Atoms .............................. 33
7.4 User-Defined Basis Sets ..................................................... 35
7.5 Integral Library Options ..................................................... 35

8 SCF : Self-Consistent Field ...................................................... 37
8.1 Overview ........................................................................ 37
8.2 ROHF ........................................................................... 38
8.3 Input Examples ................................................................. 38
8.4 SCF Module Options ........................................................ 39

9 OPT : Geometry Optimization ................................................ 46
9.1 Overview ...................................................................... 46
9.2 Input Examples ................................................................ 46
9.3 OPT Module Options ........................................................ 47

10 FREQ: Vibrational Frequencies .............................................. 51
10.1 Overview .................................................................... 51
10.2 Input Examples ................................................................ 51
10.3 FREQ Module Options .................................................... 54

11 DFOCC : Orbital-Optimized Coupled-Cluster and Moller–Plesset Perturbation Theories ............................................................... 57
<table>
<thead>
<tr>
<th>11.1 Theory</th>
<th>57</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.2 Conventional (Non-OO) Coupled-Cluster and Moller-Plesset Perturbation Theories</td>
<td>59</td>
</tr>
<tr>
<td>11.3 Input Examples</td>
<td>60</td>
</tr>
<tr>
<td>11.4 DFOCC Module Options</td>
<td>63</td>
</tr>
<tr>
<td>12 QDPT: Quasidegenerate Perturbation Theory</td>
<td>73</td>
</tr>
<tr>
<td>12.1 Input Examples</td>
<td>73</td>
</tr>
<tr>
<td>12.2 QDPT Module Options</td>
<td>74</td>
</tr>
<tr>
<td>13 EOMEE: Equation-of-Motion Coupled-Cluster</td>
<td>79</td>
</tr>
<tr>
<td>13.1 Input Examples</td>
<td>79</td>
</tr>
<tr>
<td>13.2 EOMEE Module Options</td>
<td>80</td>
</tr>
<tr>
<td>14 Fragment: Molecular Fragmentation Approaches</td>
<td>82</td>
</tr>
<tr>
<td>14.1 Systematic Molecular Fragmentation (SMF)</td>
<td>82</td>
</tr>
<tr>
<td>14.2 Input Examples</td>
<td>82</td>
</tr>
<tr>
<td>14.2.1 Serial Mode</td>
<td>82</td>
</tr>
<tr>
<td>14.2.2 Mode Sow-1</td>
<td>85</td>
</tr>
<tr>
<td>14.2.3 Mode Sow-2</td>
<td>87</td>
</tr>
<tr>
<td>14.2.4 Mode Sow-3</td>
<td>90</td>
</tr>
<tr>
<td>14.2.5 Assigning Charge to Groups</td>
<td>92</td>
</tr>
<tr>
<td>14.3 FRAGMENT Module Options</td>
<td>95</td>
</tr>
</tbody>
</table>
1 Overview

1.1 About

MacroQC is an electronic structure theory software for high-accuracy computations and large-scale chemical applications. The MacroQC software is developed and maintained by Prof. Uğur Bozkaya and his research group at the Department of Chemistry, Hacettepe University, Ankara, Turkey.

The MacroQC software avoids the usage of the conventional four-index electron repulsion integrals (ERIs). Instead, it uses the density-fitting (DF) technique \[11, 12, 13, 14, 15, 6, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28\] for all methods included. MacroQC package features embracingly rich perturbation theory and coupled-cluster theory codes, as well as their orbital-optimized variants, for energy and analytic gradient computations. The attractive feature of the MacroQC software is its molecular fragment module, which enables applications of high-accuracy quantum chemical methods to large-scale chemical systems, such as biomolecules and polymers. The developed linear-scaling techniques based on molecular fragmentation approaches enable high-accuracy computations for large molecules.

The MacroQC package originated from the modules and libraries developed by Prof. Uğur Bozkaya. The starting point of MacroQC is the development of the Tensors library, which enables numerous tensor operations. The Tensors library is developed and maintained by Bozkaya since 2013. All ab initio methods and other libraries in the MacroQC package are taking advantage of the Tensors library.

The second important milestone of MacroQC is the development of the DFOCC module, which includes many perturbation theories and coupled-cluster codes. The DFOCC module is developed and maintained by Bozkaya since 2013. Recently, Asli Ünal and Yavuz Alagöz have made contributions to the latest version of the DFOCC module. An older version of the DFOCC module is also available in the Psi4 program package (https://psicode.org).

The third important milestone of MacroQC is the development of basis set and molecular integrals libraries (Molint framework) by Uğur Bozkaya in 2019-2021 \[1\]. The Molint library provides molecular integrals and their first derivatives, over contracted Gaussian functions, for the density-fitted methods.

Development of the MacroQC software officially started in 2019. The skeleton of the software is built by Bozkaya, Ermiş, and Alagöz. The first module of MacroQC, developed by Bozkaya and Alagöz, is the SCF module. Later, Bozkaya’s DFOCC module is integrated with MacroQC by Alagöz. Further, the QDPT module of Bozkaya, which includes Quasidegenerate Perturbation Theory and FCI codes, was incorporated into the MacroQC software by Bozkaya. Bozkaya also developed EKT, FNO, and PNO codes for MacroQC.

Yavuz Alagöz and Betül Ermiş made important contributions to the MacroQC package. They are involved in the development of many core libraries. Yavuz Alagöz contributed to the SCF code and unrestricted (T) code. Alagöz developed the I/O, Timer, Memory, Master, and Options libraries of MacroQC with contributions from Ermiş and Bozkaya. Further, Alagöz converted Bozkaya’s DIIS codes into an efficient library.
Betül Ermiş made contributions to the Molecule library of MacroQC, which is responsible for parsing molecular geometry and other related issues. She also developed a geometry optimization module with Prof. Bozkaya. Ermiş significantly extends the applicability of Bozkaya’s initial Infrared code and converts it into an efficient API adding many new features to it. Betül Ermiş made substantial contributions to Bozkaya’s fragment module and converted it into an efficient API adding many new features.

Asli Ünal made contributions to the MacroQC package. Unrestricted CCSD and EOM-CCSD codes were developed by Ünal and Bozkaya. She keeps work on developing new excited-state methods for the MacroQC software. Finally, Ali Kaan Uyar has joined the MacroQC team. He is the primary developer of our web page, and he is developing local methods for MacroQC, which will be available in future releases.
1.2 Capabilities

Energy and analytic gradient methods that available in the MacroQC software are given in Table 1.

Table 1: Capabilities of the MacroQC software.

<table>
<thead>
<tr>
<th>Method</th>
<th>Energy</th>
<th>Gradient\textsuperscript{a}</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCF</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>MP2</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>MP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>MP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>CCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>CCSD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>CCSD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>CCSD(T)\textsubscript{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OMP2</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OMP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OMP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OLCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OCCD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>OCCD(T)\textsubscript{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-MP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-MP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-CCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-LCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-CCSD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-CCSD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FNO-CCSD(T)\textsubscript{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>EOM-CCSD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>EOM-CCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>EOM-OCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>QDPT2</td>
<td>UHF</td>
<td>ROHF</td>
</tr>
<tr>
<td>CAS-CI</td>
<td>UHF</td>
<td>ROHF</td>
</tr>
<tr>
<td>CIS</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>FCI</td>
<td>UHF</td>
<td>ROHF</td>
</tr>
<tr>
<td>LSSMF-SCF</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-MP2</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-MP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-MP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-CCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-LCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-CCSD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-CCSD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>LSSMF-CCSD(T)\text{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OMP2</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OMP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OMP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OLCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OCCD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-OCCD(T)\text{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-MP2.5</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-MP3</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-CCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-LCCD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-CCSD</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-CCSD(T)</td>
<td>RHF</td>
<td>UHF</td>
</tr>
<tr>
<td>LSSMF-FNO-CCSD(T)\text{A}</td>
<td>RHF</td>
<td>UHF</td>
</tr>
</tbody>
</table>

* For all gradient methods, ionization potentials via Extended Koopmans’ Theorem (EKT) are available.

## 2 Citing MacroQC

### 2.1 Overall MacroQC Package


### 2.2 Orbital-Optimized Post-Hartree-Fock Methods

#### 2.2.1 Orbital-Optimized Second-Order Perturbation Theory (OMP2)


2.2.2 Orbital-Optimized Third-Order Perturbation Theory (OMP3) and Orbital-Optimized MP2.5 (OMP2.5)


2.2.3 Orbital-Optimized Linearized Coupled-Cluster Doubles Method (OLCCCD)


2.2.4 Orbital-Optimized Coupled-Cluster Doubles Method (OCCD)


2.2.5 Orbital-Optimized Coupled-Cluster Doubles Method with Perturbative Triples [OCCD(T) and OCCD(\(T_A\))]


2.3 Perturbation Theory (PT)


2.4 Coupled-Cluster (CC)


2.5 Second-Order Quasidegenerate Perturbation Theory (QDPT2)


2.6 Extended Koopmans’ Theorem (EKT)


### 2.7 Molint Framework


### 2.8 Dipole Moments and One-Electron Properties


### 2.9 Anharmonic Force Field and IR Spectra


### 2.10 Linear-Scaling Systematic Molecular Fragmentation


### 2.11 Equation-of-Motion Coupled-Cluster

3 Download and Installation

MacroQC has an easy setup with few steps. There are several alternatives for installation methods. You can choose one of them depending on your system and personal habits. These alternatives are given below. In most cases, you need nothing more than an installer. Sometimes it’s not even necessary. Supported architectures are Linux x86_64 and OSX64.

3.1 Install with Conda (Linux and Mac OS X)

Conda is a cross-platform open-source package management system and environment management system. This option is one of the most popular and easy ways to install a package these days.

(1) If you don’t have a Conda installer, you may get it: [Conda installer](#).

(2) Run the following commands on a UNIX console.

```bash
conda install -c macroqc macroqc
```

3.2 Install with Self-Extracting Package (Linux and Mac OS X)

This method offers offline and portable installation just in one step. You need to follow the steps given below:

(1) Download MacroQC self-extracting package from this [link](#).

(2) Run installation script at the directory where the downloaded file is located (you may need root privileges):

```bash
bash install-macroqc.sh
```

3.3 Manual Installation from Zip Archive (Linux and Mac OS X)

This is another alternative way to install MacroQC.

(1) Download MacroQC Zip archive from this [link](#).

(2) Extract the Zip archive.

```bash
unzip macroqc.zip
```

(3) Copy files wherever you want.
4 Testing MacroQC

The test scripts and sample inputs can be found in the MQCDATADIR/tests directory. These tests can be run via mqc_run_tests script. The MacroQC software provides a few test options to users. all runs all tests, which may take a long time to complete, quick runs a small number of basic tests, and dfocc runs tests for the dfocc module. A summary of running test script:

- mqc_run_tests all
- mqc_run_tests quick
- mqc_run_tests scf
- mqc_run_tests ints
- mqc_run_tests dfocc
- mqc_run_tests eomee
- mqc_run_tests qdpt
- mqc_run_tests opt
- mqc_run_tests freq
- mqc_run_tests lssmf

Moreover, tests provide a good way to learning and practicing MacroQC input and output format. If a problem occurs during the tests, it can be directed to the developers. In such case please contact us from our forum (macroqc@googlegroups.com) or contact e-mail (macroqcpreg@gmail.com).

5 Running MacroQC

5.1 Setting Environmental Variables

To run MacroQC executable one needs to set MQCDATADIR and MQC_SCRATCH environmental variables. During the installation procedure MacroQC tries to automatically detect these variables. MQC_SCRATCH points to the directory where MacroQC writes temporary binary files, while MQCDATADIR points to the directory where the basis set files, test inputs, and documents are located. To set these variables, one can run the following commands on a Unix console.

Examples for bash, zsh, and sh:
5.2 Running MacroQC in Serial Mode

To run MacroQC in serial mode the user should invoke the driver of the software by simply typing `macroqc` on a Unix console. The default input file name is `input.inp` and the default output file name is `input.out`. To run an input file with a custom file name, one should type `macroqc filename.inp`. The output file will be printed as `filename.out`.

5.3 Running MacroQC in Parallel Mode Using OpenMP

MacroQC software can be run in OpenMP parallel mode. The pre-compiled binaries available at the MacroQC homepage support OpenMP parallel execution. To run the MacroQC executable with OpenMP one just need to set the environmental variables `OMP_NUM_THREADS` and `MKL_NUM_THREADS` to the number of cores that are requested. For example, in a BASH shell, one should set `export OMP_NUM_THREADS=8` and `export MKL_NUM_THREADS=8` to use 8 cores. Then the MacroQC program should be executed as described earlier.

6 Basic Input Format

MacroQC execution is controlled by a user-prepared input file. If the input file name is not specified on the command line, the data is read from the default input file (`input.inp`), and program results are printed to a standard output file (`input.out`). If the input file name is specified, the output file name is generated from the input file name removing any trailing suffix, and appending `.out`. If the output file already exists, MacroQC overwrites it.

MacroQC execution is controlled by a set of data and options in the input file. In general, each input record begins with a keyword, which is followed by keyword values. The command-line interface requires a MacroQC input file which is simply an ASCII text file. This input file can be created using your favorite editor (e.g., vim, emacs, jot, etc.) following the basic steps outlined in the next few chapters.

MacroQC input mechanism uses a series of section (block) keywords. The MacroQC program searches the input file for supported section names. When MacroQC finds a section name, it then reads supported keywords belong to the section. Each input section starts with `$set_section` and terminates with `$end_section`. For example, `scf` module options can be given in the `scf` block, which starts with `$set_scf` and terminates with `$end_scf`. A short description of all MacroQC keywords is provided in Table 2 (pp 16).
Table 2: The list of MacroQC input sections. The molecule section is only required for all jobs.

<table>
<thead>
<tr>
<th>Section Start</th>
<th>Section End</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$set_molecule</td>
<td>$end_molecule</td>
<td>Contains charge, multiplicity, and the molecular coordinate input (mandatory).</td>
</tr>
<tr>
<td>$setGlobals</td>
<td>$endGlobals</td>
<td>Global options.</td>
</tr>
<tr>
<td>$set_basis</td>
<td>$end_basis</td>
<td>Assign primary basis sets for individual atoms.</td>
</tr>
<tr>
<td>$setAuxBasisScf</td>
<td>$endAuxBasisScf</td>
<td>Assign JK-Fit type auxiliary basis sets for individual atoms.</td>
</tr>
<tr>
<td>$setAuxBasisCorr</td>
<td>$endAuxBasisCorr</td>
<td>Assign RI-type auxiliary basis sets for individual atoms.</td>
</tr>
<tr>
<td>$setSadAuxBasis</td>
<td>$endSadAuxBasis</td>
<td>Assign the auxiliary basis set for the SAD guess.</td>
</tr>
<tr>
<td>$setOpt</td>
<td>$endOpt</td>
<td>Options for the opt module.</td>
</tr>
<tr>
<td>$setFreq</td>
<td>$endFreq</td>
<td>Options for the freq module.</td>
</tr>
<tr>
<td>$setScf</td>
<td>$endScf</td>
<td>Options for the scf module.</td>
</tr>
<tr>
<td>$setDfocc</td>
<td>$endDfocc</td>
<td>Options for the dfocc module.</td>
</tr>
<tr>
<td>$setQdpt</td>
<td>$endQdpt</td>
<td>Options for the qdpt module.</td>
</tr>
<tr>
<td>$setEomee</td>
<td>$endEomee</td>
<td>Options for the eomee module.</td>
</tr>
<tr>
<td>$setLssmf</td>
<td>$endLssmf</td>
<td>Options for the fragment module.</td>
</tr>
<tr>
<td>$setFragExternal</td>
<td>$endFragExternal</td>
<td>Options for the external software used for fragment module.</td>
</tr>
<tr>
<td>$setDgExternal</td>
<td>$endDgExternal</td>
<td>Options for the external software used for freq module.</td>
</tr>
</tbody>
</table>

* Note that users can enter keyword sections in any order.
* Note that after $set_molecule user may add a name such as $set_molecule water. This feature is valid only for the molecule section.
* Note that molecular geometry input must be given in Cartesian coordinates, and atom symbols should be used in the geometry section.
6.1 Input Examples

*Single-point energy computation at the rhf/cc-pvdz level:

```plaintext
# This line is a comment line
$set_molecule H2O
  0 1
  O  0.00000  0.00000  -0.06577
  H  0.00000  -0.75906   0.52195
  H  0.00000   0.75906   0.52195
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  method scf
  reference rhf
  jobtype energy
  memory 2000
$end_globals
```

6.2 Ghost Atoms

Writing the @ symbol before the atom symbol of any atom in the molecule section will make that atom a ghost atom. Ghost atoms possess basis functions but they do not have electrons or nuclear charges. The ghost atoms can only be used for energy computations.

*Single-point energy computation at the rhf/cc-pvdz level:

```plaintext
$set_molecule H2O-NH3
  0 1
  O  0.00000  0.00000  -0.06577
  H  0.00000  -0.75906   0.52195
  H  0.00000   0.75906   0.52195
  @N  0.00000   0.01436   1.46454
  @H  0.00000  -0.98104   1.65344
  @H  -0.81348   0.39876   1.92934
  @H   0.81348   0.39876   1.92934
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  method scf
  reference rhf
  jobtype energy
```
6.3 External Charges

The format for external charges is the EC symbol, the Cartesian coordinates, followed by the charge. External charges are also shifted to the center of mass with the molecule.

*Single-point energy computation at the rhf/cc-pvdz level:

```plaintext
$set_molecule
1 1
O  1.46710223 -0.96026401 -0.84235688
H  2.38494062 -0.65229087 -0.74535997
H  1.50141665 -1.75285797 -1.39080538
O -1.45264971  0.60446441 -1.14352866
H -1.47721698  1.15725057 -1.93346483
H -2.37258328  0.34732430 -0.95837892
O  1.14694134  1.35761780  0.78995323
H  0.96540138  2.12759124  1.34163941
H  2.11027535  1.33621950  0.65502051
O -1.16127440 -1.00259910  1.19567449
H -0.98941492 -1.53327044  1.98245878
H -2.12261877 -1.03132143  1.0484012
O  3.74848604  0.57050669 -0.07526597
H  4.25830799  1.09057781 -0.71369648
H  4.41479872  0.22966949  0.53942387
O -3.74861116 -0.56969516  0.07555459
H -4.25143300 -1.27985843 -0.34967876
H -4.42204079 -0.03797115  0.52471820
EC  0.0003690  -0.00022249 -0.00011859  1.0
$end_molecule

$setGlobals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  method scf
  reference rhf
  jobtype energy
$endGlobals
```

6.4 Multiple Jobs

Multiple jobs can be combined into a single input file. The @@@ string line is used to separate jobs. All results are appended to a single output file. If you want to use the same options
for different geometries, you should give the options in the first job. It is also possible to run completely independent jobs in the same input.

*Single-point energy computation at the mp2/cc-pvdz level for multiple molecular geometries using a single option block:

```plaintext
$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  method scf
  reference rhf
  jobtype energy
$end Globals

$set_molecule H2O
  0 1
  O 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

@@@

$set_molecule CH4
  0 1
  C 1.06474 0.20611 0.30975
  H 0.84383 0.16523 1.38297
  H 0.98386 1.24826 -0.00369
  H 0.38021 -0.42906 -0.24139
  H 2.07622 -0.15282 0.18373
$end_molecule

@@@

*Completely independent jobs in the same input.

$set_molecule H2O
  0 1
  O 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$setGlobals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  method scf
```

19
6.5 Global Options

- **FREEZE_CORE**
  Do apply frozen core approximation for post-Hartree–Fock computations? See frozen core table for individual atoms.
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **MOL_FRAG**
  Do use molecular fragmentation techniques? See fragment module options if this option is set to TRUE.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **NOCOM**
  Do not shift molecular geometry to the center of mass?
TYPE: Boolean
DEFAULT: FALSE
OPTIONS: TRUE, FALSE

• **NOREORIENT**
  Do not reorient molecular geometry?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

• **PUREAM**
  Do use pure angular momentum functions in molecular integral evaluations? The molint library is optimized for spherical functions. The usage of Cartesian functions is experimental. Hence, we recommend using the default option.
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

• **TEST_MODE**
  Activates test mode.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **FINDIF_POINTS**
  The number of points used in finite difference formulas.
  TYPE: Integer
  DEFAULT: 3
  OPTIONS: 3, 5

• **MEMORY**
  Amount of total memory supplied to the software in MB.
  TYPE: Integer
  DEFAULT: 1000
  OPTIONS: n User-defined value.

• **NUM_FROZEN_DOCC**
  The number of frozen doubly occupied orbitals to be used in post-Hartree–Fock computations. This option overwrites the FREEZE_CORE option.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.

• **NUM_FROZEN_VIR**
  Number of frozen virtuals to be used in post-Hartree–Fock computations.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.
• **PRINT_LEVEL**
  This option controls the amount of information to print to the output file.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: $n$ User-defined value.

• **AUX_BASIS_CORR**
  Auxiliary basis set to be used in electron correlation computations.
  TYPE: String
  DEFAULT: CC-PVDZ-RI
  OPTIONS: User defined basis set.

• **AUX_BASIS_SCF**
  Auxiliary basis set to be used in SCF computations.
  TYPE: String
  DEFAULT: CC-PVDZ-JKFIT
  OPTIONS: User defined basis set.

• **BASIS**
  Primary basis set to be used in computations.
  TYPE: String
  DEFAULT: CC-PVDZ
  OPTIONS: User defined basis set.

• **DERIVE_TYPE**
  Type of derivatives to be used.
  TYPE: String
  DEFAULT: ANALYTIC
  OPTIONS: ANALYTIC, NUMERIC

• **FINDIF_METHOD**
  The method used in FINDIF formulas.
  TYPE: String
  DEFAULT: GRAD
  OPTIONS: GRAD, ENERGY

• **FRAG_METHOD**
  The molecular fragmentation method will be used. For now, MacroQC software has only the LSSMF approach. However, more methods are coming soon.
  TYPE: String
  DEFAULT: LSSMF
  OPTIONS: LSSMF

• **GEOM_UNITS**
  The unit of molecular geometry.
  TYPE: String
  DEFAULT: ANG
  OPTIONS: ANG, BOHR
• JOBTYPE
Type of job that requested from the software.
TYPE: String
DEFAULT: ENERGY
OPTIONS: ENERGY, GRAD, OPT, FREQ, INPUT
  - ENERGY: Single-point energy job.
  - GRAD: Analytic gradients job.
  - OPT: Geometry optimization job.
  - FREQ: Vibrational frequency job.
  - INPUT: Checks the syntax, charge, and multiplicity without any computations.

• REFERENCE
The reference wave function for SCF computations.
TYPE: String
DEFAULT: RHF
OPTIONS: RHF, UHF, ROHF
  - RHF: Restricted closed-shell Hartree–Fock.
  - UHF: Unrestricted Hartree–Fock.
  - ROHF: Restricted open-shell Hartree–Fock.

• METHOD
A Quantum chemical method will be used.
TYPE: String
DEFAULT: SCF
OPTIONS: SCF, QCHF, MP2, OMP2, MP2.5, OMP2.5, MP3, OMP3, CIS, CCD, OCCD, OCCD(T), OCCD(AT), LCCD, OLCCD, CCSD, CCSD(T), CCSD(AT), QDPT2, CAS-CI, FCI, EOM-CCSD
  - SCF: Self-consistent field method.
  - MP2: Second-order Møller–Plesset perturbation theory.
  - OMP2: Orbital-optimized second-order Møller–Plesset perturbation theory.
  - MP2.5: The MP2.5 model.
  - OMP2.5: Orbital-optimized MP2.5 model.
  - MP3: Third-order Møller–Plesset perturbation theory.
  - OMP3: Orbital-optimized third-order Møller–Plesset perturbation theory.
  - CIS: Configuration interaction singles.
  - CCD: Coupled-cluster doubles.
  - OCCD: Orbital-optimized coupled-cluster doubles.
  - OCCD(T): Orbital-optimized coupled-cluster doubles with perturbative triples.
OCCD(AT): Orbital-optimized coupled-cluster doubles with asymmetric perturbative triples.
- LCCD: Linearized coupled-cluster doubles.
- OLCCD: Orbital-optimized linearized coupled-cluster doubles.
- CCSD: Coupled-cluster singles and doubles.
- CCSD(T): Coupled-cluster singles and doubles with perturbative triples.
- CCSD(AT): Coupled-cluster singles and doubles with asymmetric perturbative triples.
- CAS-CI: Complete active space configuration interaction.
- FCI: Full configuration interaction.
- EOM-CCSD: Equation-of-motion coupled-cluster singles and doubles.

6.6 Default Values for the Frozen Core Approximation

In electron correlation computations only the valence electrons are generally considered. The remaining electrons are kept frozen, which defines a frozen core. Table 3 presents default values of frozen core orbitals for post-HF computations. These values can be overwritten by user-defined values using the NUM_FROZEN_DOCC option.

Table 3: Default values for number of frozen core orbitals.

<table>
<thead>
<tr>
<th>0</th>
<th>H</th>
<th>0</th>
<th>Be</th>
<th>1</th>
<th>B</th>
<th>1</th>
<th>C</th>
<th>1</th>
<th>N</th>
<th>1</th>
<th>O</th>
<th>1</th>
<th>F</th>
<th>1</th>
<th>Ne</th>
</tr>
</thead>
<tbody>
<tr>
<td>Li</td>
<td>5</td>
<td>Ca</td>
<td>5</td>
<td>Sc</td>
<td>5</td>
<td>Ti</td>
<td>5</td>
<td>V</td>
<td>5</td>
<td>Cr</td>
<td>5</td>
<td>Mn</td>
<td>5</td>
<td>Fe</td>
<td>5</td>
</tr>
<tr>
<td>Na</td>
<td>1</td>
<td>Mg</td>
<td>1</td>
<td>Al</td>
<td>1</td>
<td>Si</td>
<td>1</td>
<td>P</td>
<td>1</td>
<td>S</td>
<td>1</td>
<td>Cl</td>
<td>1</td>
<td>Br</td>
<td>1</td>
</tr>
<tr>
<td>K</td>
<td>9</td>
<td>Ca</td>
<td>14</td>
<td>Y</td>
<td>14</td>
<td>Zr</td>
<td>14</td>
<td>Nb</td>
<td>14</td>
<td>Mo</td>
<td>14</td>
<td>Te</td>
<td>14</td>
<td>Ru</td>
<td>14</td>
</tr>
<tr>
<td>Rb</td>
<td>18</td>
<td>Ba</td>
<td>23</td>
<td>Lu</td>
<td>23</td>
<td>Hf</td>
<td>23</td>
<td>Ta</td>
<td>23</td>
<td>W</td>
<td>23</td>
<td>Re</td>
<td>23</td>
<td>Os</td>
<td>23</td>
</tr>
<tr>
<td>Cs</td>
<td>34</td>
<td>Ra</td>
<td>50</td>
<td>Rf</td>
<td>50</td>
<td>Db</td>
<td>50</td>
<td>Sg</td>
<td>50</td>
<td>Bh</td>
<td>50</td>
<td>Hs</td>
<td>50</td>
<td>Mt</td>
<td>50</td>
</tr>
<tr>
<td>Fr</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
</tbody>
</table>

Please note that Dunning’s cc-pVXZ (X=2-6) basis sets are designed for frozen core computations. Hence, in the case of all-electron computations, it is better to use the cc-pCVXZ or cc-pwCVXZ (X=2-6) sets instead.
7 Basis Sets

7.1 Overview

A basis set is a set of functions used to generate molecular orbitals by a linear combination of basis functions. MacroQC uses atom-centered orbitals as basis sets, in short, atomic orbitals (AO). MacroQC employs contracted Gaussian functions as basis functions. By default, MacroQC generates spherical functions, more specifically real-valued solid-harmonics. The usage of Cartesian functions are also supported for experimental purposes.

The general form of unnormalized primitive Cartesian Gaussian type orbitals (GTOs) centered at \( \mathbf{R}_A \) can be written as

\[
G_{ijk}(a, \mathbf{r}_A) = (x - A_x)^i (y - A_y)^j (z - A_z)^k e^{-a r^2_A}, \tag{1}
\]

where \( i, j, k \) are the \( x, y, z \) components of the angular momentum, respectively, and \( \mathbf{r} \) is the position vector.

A real-valued spherical GTO with quantum numbers \( l \) and \( m \), with exponent \( a \), centered at \( \mathbf{A} \) is given by,

\[
G_{lm}(\mathbf{r}, a, \mathbf{A}) = S_{lm}(x_A, y_A, z_A)e^{-ar^2_A}, \tag{2}
\]

where \( S_{lm}(\mathbf{r}_A) \) are the real solid-harmonics [1, 4]. The spherical GTOs can be obtained from the Cartesian GTOs by a simple transformation [4, 5].

Contracted Gaussian type orbitals (CGTOs) can be written as

\[
G_{\mu lm}(\mathbf{r}, a, \mathbf{A}) = \sum_p d_{\mu p} G_{lm}(\mathbf{r}, a_p, \mathbf{A}), \tag{3}
\]

where \( d_{\mu p} \) is the contraction coefficient.

7.2 Built-In Basis Sets

The built-in primary basis sets and their default JK-Fit and RI-Fit auxiliary basis sets in the MacroQC software are presented in Table 5. In the input file, the primary basis set is defined with the basis option, while JK-Fit [6] and RI-Fit [7] basis sets are defined with aux_basis_scf and aux_basis_corr options, respectively. All these options should be written in the globals block. Hence, the user may change default auxiliary basis sets with these options. Further, in the input file, the names given in the Basis column of Table
should be written. For example, for the cc-pV(D+d)Z basis set, one needs to set basis cc-pv_dpd_z.

Angular momentum convention that used in the basis library of the MacroQC program is reported in Table 4.

<table>
<thead>
<tr>
<th>$L$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Orbital</td>
<td>s</td>
<td>p</td>
<td>d</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>i</td>
<td>k</td>
<td>l</td>
<td>m</td>
<td>n</td>
</tr>
</tbody>
</table>

Table 4: Angular momentum convention used in MacroQC.
<table>
<thead>
<tr>
<th>Basis Family</th>
<th>Basis</th>
<th>JK Auxiliary Basis</th>
<th>MP2 (RI) Auxiliary Basis</th>
</tr>
</thead>
<tbody>
<tr>
<td>cc-pVDZ</td>
<td>cc-pvdz</td>
<td>cc-pvdz-jkfit</td>
<td>cc-pvdz-ri</td>
</tr>
<tr>
<td>cc-pV(D+d)Z</td>
<td>cc-pv_dpd_z</td>
<td>cc-pvdz-jkfit</td>
<td>cc-pvdz-ri</td>
</tr>
<tr>
<td>cc-pCVZ</td>
<td>cc-pcvd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cc-pCV(D+d)Z</td>
<td>cc-pcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cc-pwCVDZ</td>
<td>cc-pwcvd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pVDZ</td>
<td>aug-cc-pvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pV(D+d)Z</td>
<td>aug-cc-pvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pCVZ</td>
<td>aug-cc-pcvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pCV(D+d)Z</td>
<td>aug-cc-pcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pVDZ</td>
<td>heavy-aug-cc-pvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pV(D+d)Z</td>
<td>heavy-aug-cc-pv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pCVZ</td>
<td>heavy-aug-cc-pcvd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pCV(D+d)Z</td>
<td>heavy-aug-cc-pcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pwCVDZ</td>
<td>heavy-aug-cc-pwcvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pwCV(D+d)Z</td>
<td>heavy-aug-cc-pwcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pVDZ</td>
<td>jun-cc-pvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pV(D+d)Z</td>
<td>jun-cc-pv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pCVZ</td>
<td>jun-cc-pcvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pCV(D+d)Z</td>
<td>jun-cc-pcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pwCVDZ</td>
<td>jun-cc-pwcvd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>jun-cc-pwCV(D+d)Z</td>
<td>jun-cc-pwcv_dpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d-aug-cc-pVDZ</td>
<td>d-aug-cc-pvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d-aug-cc-pCVZ</td>
<td>d-aug-cc-pcvdz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>d-aug-cc-pwCVDZ</td>
<td>d-aug-cc-pwcvd</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cc-pVTZ</td>
<td>cc-pvtz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cc-pV(T+d)Z</td>
<td>cc-pv_tpd_z</td>
<td>cc-pvtz-jkfit</td>
<td>cc-pvtz-ri</td>
</tr>
<tr>
<td>cc-pCVTZ</td>
<td>cc-pcvt</td>
<td>cc-pcvt-jkfit</td>
<td>cc-pcvt-ri</td>
</tr>
<tr>
<td>cc-pCV(T+d)Z</td>
<td>cc-pcv_tpd_z</td>
<td>cc-pcvt-jkfit</td>
<td>cc-pcvt-ri</td>
</tr>
<tr>
<td>cc-pwCVTZ</td>
<td>cc-pwcvtz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>cc-pwCV(T+d)Z</td>
<td>cc-pwcv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pVTZ</td>
<td>aug-cc-pvtz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pV(T+d)Z</td>
<td>aug-cc-pv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pCVTZ</td>
<td>aug-cc-pcvt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pCV(T+d)Z</td>
<td>aug-cc-pcv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pwCVTZ</td>
<td>aug-cc-pwcvt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>aug-cc-pwCV(T+d)Z</td>
<td>aug-cc-pwcv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pVTZ</td>
<td>heavy-aug-cc-pvtz</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pV(T+d)Z</td>
<td>heavy-aug-cc-pv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pCVTZ</td>
<td>heavy-aug-cc-pcvt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pCV(T+d)Z</td>
<td>heavy-aug-cc-pcv_tpd_z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>heavy-aug-cc-pwCVTZ</td>
<td>heavy-aug-cc-pwcvt</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configuration</td>
<td>28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------</td>
<td>----</td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>heavy-aug-cc-pwCV(T+d)Z</code></td>
<td><code>heavy-aug-cc-pwcv_tpd_z</code></td>
<td><code>heavy-aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>jun-cc-pVTZ</code></td>
<td><code>jun-cc-pv_td</code></td>
<td><code>jun-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>jun-cc-pV(T+d)Z</code></td>
<td><code>jun-cc-pcv_tpd_z</code></td>
<td><code>jun-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>jun-cc-pCV(T+d)Z</code></td>
<td><code>jun-cc-pwcv_tpd_z</code></td>
<td><code>jun-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pVTZ</code></td>
<td><code>may-cc-pv_td</code></td>
<td><code>may-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pV(T+d)Z</code></td>
<td><code>may-cc-pcv_tpd_z</code></td>
<td><code>may-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pCV(T+d)Z</code></td>
<td><code>may-cc-pwcv_tpd_z</code></td>
<td><code>may-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pwCV(T+d)Z</code></td>
<td><code>may-cc-pwcv_tpd_z</code></td>
<td><code>may-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>d-aug-cc-pVTZ</code></td>
<td><code>d-aug-cc-pvqz</code></td>
<td><code>d-aug-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>d-aug-cc-pCV(T+d)Z</code></td>
<td><code>d-aug-cc-pvqz</code></td>
<td><code>d-aug-cc-pcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>ce-pVTZ-DK</code></td>
<td><code>ce-pvqz</code></td>
<td><code>ce-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>ce-pCV(T+d)Z</code></td>
<td><code>ce-pvqz</code></td>
<td><code>ce-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pVTZ-DK</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pwCV(T+d)Z</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>aug-cc-pVTZ-DK</code></td>
<td><code>aug-cc-pvqz</code></td>
<td><code>aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>d-aug-cc-pVTZ-DK</code></td>
<td><code>d-aug-cc-pvqz</code></td>
<td><code>d-aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>heavy-aug-cc-pVTZ-DK</code></td>
<td><code>heavy-aug-cc-pvqz</code></td>
<td><code>heavy-aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>heavy-aug-cc-pCV(T+d)Z</code></td>
<td><code>heavy-aug-cc-pcwcvqz</code></td>
<td><code>heavy-aug-cc-pcwcvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pVQZ</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pV(Q+d)Z</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pCV(Q+d)Z</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>cc-pwCVQZ</code></td>
<td><code>cc-pvqz</code></td>
<td><code>cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>aug-cc-pVQZ</code></td>
<td><code>aug-cc-pvqz</code></td>
<td><code>aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>aug-cc-pCVQZ</code></td>
<td><code>aug-cc-pvqz</code></td>
<td><code>aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>aug-cc-pwCVQZ</code></td>
<td><code>aug-cc-pvqz</code></td>
<td><code>aug-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pVQZ</code></td>
<td><code>may-cc-pvqz</code></td>
<td><code>may-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pCVQZ</code></td>
<td><code>may-cc-pvqz</code></td>
<td><code>may-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td><code>may-cc-pwCVQZ</code></td>
<td><code>may-cc-pvqz</code></td>
<td><code>may-cc-pvqz</code></td>
<td></td>
</tr>
<tr>
<td>Configuration</td>
<td>STO-3G</td>
<td>def2-universal-jkfit</td>
<td>def2-svp-ri</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------</td>
<td>----------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>apr-cc-pwCV(5+d)Z</td>
<td>apr-cc-pwcv_5pd_z</td>
<td>apr-cc-pv5z-jkfit</td>
<td>apr-cc-pwcv5z-ri</td>
</tr>
<tr>
<td>mar-cc-pV5Z</td>
<td>mar-cc-pv5z</td>
<td>mar-cc-pv5z-jkfit</td>
<td>mar-cc-pv5z-ri</td>
</tr>
<tr>
<td>mar-cc-pV(5+d)Z</td>
<td>mar-cc-pv_5pd_z</td>
<td>mar-cc-pcv5z-jkfit</td>
<td>mar-cc-pv5z-ri</td>
</tr>
<tr>
<td>mar-cc-pCV5Z</td>
<td>mar-cc-pcv5z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>mar-cc-pwCV(5+d)Z</td>
<td>mar-cc-pwcv5z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>d-aug-cc-pV5Z</td>
<td>d-aug-cc-pv5z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>d-aug-cc-pCV5Z</td>
<td>d-aug-cc-pcv5z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pV5Z-DK</td>
<td>cc-pv5z-dk</td>
<td>cc-pv6z-ri</td>
<td>cc-pv6z-ri</td>
</tr>
<tr>
<td>cc-pwCV5Z-DK</td>
<td>cc-pwcv5z-dk</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pV5Z-DK</td>
<td>aug-cc-pv5z-dk</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pCV5Z-DK</td>
<td>aug-cc-pcv5z-dk</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>heavy-aug-cc-pV5Z-DK</td>
<td>heavy-aug-cc-pv5z-dk</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>heavy-aug-cc-pCV5Z-DK</td>
<td>heavy-aug-cc-pcv5z-dk</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pV(6+d)Z</td>
<td>cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pCV6Z</td>
<td>cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pCV(6+d)Z</td>
<td>cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pV6Z</td>
<td>aug-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pV(6+d)Z</td>
<td>aug-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pCV6Z</td>
<td>aug-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>heavy-aug-cc-pV6Z</td>
<td>heavy-aug-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>heavy-aug-cc-pCV6Z</td>
<td>heavy-aug-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pV(6+d)Z</td>
<td>cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>cc-pCV(6+d)Z</td>
<td>cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>aug-cc-pCV(6+d)Z</td>
<td>aug-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>jun-cc-pV6Z</td>
<td>jun-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>jun-cc-pV(6+d)Z</td>
<td>jun-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>jun-cc-pCV6Z</td>
<td>jun-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>jun-cc-pCV(6+d)Z</td>
<td>jun-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>may-cc-pV6Z</td>
<td>may-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>may-cc-pV(6+d)Z</td>
<td>may-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>may-cc-pCV6Z</td>
<td>may-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>may-cc-pCV(6+d)Z</td>
<td>may-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>apr-cc-pV6Z</td>
<td>apr-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>apr-cc-pV(6+d)Z</td>
<td>apr-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>apr-cc-pCV6Z</td>
<td>apr-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>apr-cc-pCV(6+d)Z</td>
<td>apr-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>mar-cc-pV6Z</td>
<td>mar-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>mar-cc-pV(6+d)Z</td>
<td>mar-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>mar-cc-pCV6Z</td>
<td>mar-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>mar-cc-pCV(6+d)Z</td>
<td>mar-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>feb-cc-pV6Z</td>
<td>feb-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>feb-cc-pV(6+d)Z</td>
<td>feb-cc-pv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>feb-cc-pCV6Z</td>
<td>feb-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>feb-cc-pCV(6+d)Z</td>
<td>feb-cc-pcv_6pd_z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>d-aug-cc-pV6Z</td>
<td>d-aug-cc-pv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>d-aug-cc-pCV6Z</td>
<td>d-aug-cc-pcv6z</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>STO-3G</td>
<td>sto-3g</td>
<td>def2-universal-jkfit</td>
<td>def2-svp-ri</td>
</tr>
<tr>
<td>Level</td>
<td>Method</td>
<td>Basis Set</td>
<td>Frozen Core</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------</td>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>STO-6G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>3-21G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31G(d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31G(d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31G*</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31G**</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31+G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31+G(d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31+G(p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31+G*</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-31+G**</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G*</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G**</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,3p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G(d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G(d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G*</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G**</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2d,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(2df,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,2p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311G(3df,3p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G(d)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G(d,p)</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G*</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
<tr>
<td>6-311+G**</td>
<td>def2-universal-jkfit</td>
<td>6-31g</td>
<td></td>
</tr>
</tbody>
</table>

**Notes:**
- \( ZP_\) and \( T_\) are not explicitly listed in the table.
- \( \text{E_Total} \) includes both \( ZP_\) and \( T_\) contributions.
- Spin multiplicity is indicated by \( m_s \) values ranging from 0 to 3.
- Description includes specific level designations and basis set enhancements.
7.3 Assigning Basis Sets for Individual Atoms

Different basis sets can be assigned to individual atoms in the `BASIS`, `AUX_BASIS_SCF`, `AUX_BASIS_CORR`, and `SAD_AUX_BASIS` blocks. For example:

```plaintext
$set_molecule
  0 1
  C  -1.948844  0.619359  -0.000000
  C   -0.016018  0.608680   0.000000
  H   0.336565  -0.131599  -0.687440
  H   0.345972   1.571195  -0.295683
  H   0.339394   0.380532   0.983124
  H  -2.301427   1.359637   0.687439
  H  -2.304256   0.847506  -0.983124
  H  -2.310834  -0.343156   0.295683
 Zn  -0.186875   3.128829   0.000000
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method mp2
  reference uhf
  jobtype energy
$end Globals

$set_basis
  Zn  def2-qzvp
$end_basis

$set_aux_basis_scf
```
Zn_def2-universal-jkfit
$end_aux_basis_scf

$set_aux_basis_corr
    Zn_def2-qzvp-ri
$end_aux_basis_corr

The relevant part of the output file for this example:

```
CC-PVDZ primary basis set file will be read for the C atom...
CC-PVDZ primary basis set file will be read for the C atom...
CC-PVDZ primary basis set file will be read for the H atom...
CC-PVDZ primary basis set file will be read for the H atom...
CC-PVDZ primary basis set file will be read for the H atom...
CC-PVDZ primary basis set file will be read for the H atom...
CC-PVDZ primary basis set file will be read for the H atom...
DEF2-QZVP primary basis set file will be read for the ZN atom...
Nmaxthread: 8
Nthread: 1
Nshell: 56
Lmax: 4
Npmax: 11
Nao: 142
Ndocc: 24
Nsocc: 0
NoccA: 24
NoccB: 24
Primary basis set info has been read.

CC-PVDZ-JKFIT aux basis set file will be read for the C atom...
CC-PVDZ-JKFIT aux basis set file will be read for the C atom...
CC-PVDZ-JKFIT aux basis set file will be read for the H atom...
CC-PVDZ-JKFIT aux basis set file will be read for the H atom...
CC-PVDZ-JKFIT aux basis set file will be read for the H atom...
CC-PVDZ-JKFIT aux basis set file will be read for the H atom...
CC-PVDZ-JKFIT aux basis set file will be read for the H atom...
DEF2-UNIVERSAL-JKFIT aux basis set file will be read for the ZN atom...
Nshell_aux: 102
Naux: 278
Aux basis set info has been read.
```

...
7.4 User-Defined Basis Sets

MacroQC uses Gaussian format for basis sets. For example, the \texttt{cc-pvdz} basis set for the H atom:

\begin{verbatim}
H 0
S 3 1.00
   13.0100000 0.0196850
   1.9620000 0.1379770
   0.4446000 0.4781480
S 1 1.00
   0.1220000 1.0000000
P 1 1.00
   0.7270000 1.0000000
****
\end{verbatim}

To submit a general use-defined basis to MacroQC one needs to prepare a \texttt{.gbs} file in which the basis set should be given in the Gaussian format. For example, if one prepares a basis set file titled \texttt{mybasis.gbs}, this file should be placed in the \texttt{MQCIDIR/macroqc/share/basis} directory. \texttt{MQCIDIR} is the directory where MacroQC is installed. For example: \texttt{/opt/macroqc}.

Many basis sets can be obtained from the \texttt{Basis Set Exchange} website: \url{http://basissetexchange.org}

7.5 Integral Library Options

- **PRINT_LEVEL**
  The amount of information to print to the output file.

35
• **MAX_AM**
  Maximum angular momentum value for the primary basis set.
  TYPE: Integer
  DEFAULT: 7
  OPTIONS: $n$ User-defined value.

• **MAX_AM_AUX**
  Maximum angular momentum value for the auxiliary basis set.
  TYPE: Integer
  DEFAULT: 7
  OPTIONS: $n$ User-defined value.

• **TAYLOR_ORDER**
  Order of Taylor series expansion to compute Boys function.
  TYPE: Integer
  DEFAULT: 6
  OPTIONS: $n$ User-defined value.

• **BOYS_MAXITER**
  Maximum number of iterations for the Boys grid.
  TYPE: Integer
  DEFAULT: 10000
  OPTIONS: $n$ User-defined value.

• **X_MAX**
  The maximum value of $x$ in the Boys grid.
  TYPE: Double
  DEFAULT: 30.0
  OPTIONS: $n$ User-defined value.

• **INTEGRAL_CUTOFF**
  The cutoff value for numerical procedures.
  TYPE: Double
  DEFAULT: 1.0e-10
  OPTIONS: $n$ User-defined value.

• **BOYS_CUTOFF**
  The cutoff value for numerical procedures in the Boys grid.
  TYPE: Double
  DEFAULT: 1.0E-15
  OPTIONS: $n$ User-defined value.

• **BOYS_ZERO**
  The threshold value for numerical procedures in the Boys grid.
TYPE: Double
DEFAULT: 1.0E-6
OPTIONS: n User-defined value.

- **DELTA_X**
  Step size in the Boys grid.
  TYPE: Double
  DEFAULT: 0.1
  OPTIONS: n User-defined value.

- **3INDEX_ALGORITHM**
  The algorithm used for computations of 3-index ERIs.
  TYPE: String
  DEFAULT: AUTO
  OPTIONS: AUTO, OS1, OS2, MD4

- **BOYS_METHOD**
  The algorithm is used to compute grid values.
  TYPE: String
  DEFAULT: GHP
  OPTIONS: GHP, SAUNDERS

8 **SCF : Self-Consistent Field**

**Code Authors:** U. Bozkaya and Y. Alagöz.

8.1 **Overview**

`MacroQC` performs self-consistent field computations for restricted, unrestricted, and restricted open-shell Hartree–Fock (RHF, UHF, and ROHF) references. `MacroQC` provides several different algorithms to obtain the HF solution, such as the conventional Roothan-Hall procedure, approximately second-order algorithm (ASO-SCF) [9], and the quadratically convergent SCF algorithm (QC-SCF). The `scf` module employs several initial guesses, such as the generalized Wolfsberg-Helmholtz (GWH) [10], superposition of atomic density (SAD), and the core Hamiltonian. If there is enough memory, `MacroQC` uses an incore JK-algorithm to build the Fock matrix. In the case of limited memory, one may employ an integral-direct approach [1].
8.2 ROHF

The ROHF Fock matrix can be written as follows:

\[
F = \begin{pmatrix}
F_{cc} & F_{co} & F_{cv} \\
F_{oc} & F_{oo} & F_{ov} \\
F_{vc} & F_{vo} & F_{vv}
\end{pmatrix}
\] (4)

where \(c, o, v\) are closed, open, and virtual parts. In general, the ROHF effective Fock matrix can be written as follows:

\[
F = \begin{pmatrix}
A_{cc}F^\alpha + B_{cc}F^\beta & F^\beta & \frac{1}{2}(F^\alpha + F^\beta) \\
F^\beta & A_{oo}F^\alpha + B_{oo}F^\beta & F^\alpha \\
\frac{1}{2}(F^\alpha + F^\beta) & F^\alpha & A_{vv}F^\alpha + B_{vv}F^\beta
\end{pmatrix}
\] (5)

In MacroQC there are several options available for ROHF coefficients (Table 6), such as Guest and Saunders [29], Roothaan [30], Davidson [31], Binkley, Pople, Dobosh [32], McWeeny and Diercksen [33], Faegri and Manne [34], Plakhutin, Gorelik, and Breslavskaya [35], and GAMESS GVB program [36].

| ROHF coefficients available in MacroQC. |
|---|---|---|---|---|---|---|
| **Acc** | **Bcc** | **Aoo** | **Boo** | **Avv** | **Bvv** |
| Guest and Saunders | 1/2 | 1/2 | 1/2 | 1/2 | 1/2 | 1/2 |
| Roothaan | -1/2 | 3/2 | 1/2 | 1/2 | 3/2 | -1/2 |
| Davidson | 1/2 | 1/2 | 1 | 0 | 1 | 0 |
| Binkley, Pople, Dobosh | 1/2 | 1/2 | 1 | 0 | 0 | 1 |
| McWeeny and Diercksen | 1/3 | 2/3 | 1/3 | 1/3 | 2/3 | 1/3 |
| Faegri and Manne | 1/2 | 1/2 | 1 | 0 | 1/2 | 1/2 |
| Plakhutin, Gorelik, and Breslavskaya | 0 | 1 | 1 | 0 | 1 | 0 |
| GAMESS GVB program | 1/2 | 1/2 | 1/2 | 0 | 1/2 | 1/2 |
| Pure Alpha | 1 | 0 | 0 | 0 | 1 | 0 |
| Pure Beta | 0 | 1 | 0 | 1 | 0 | 1 |

8.3 Input Examples

*Single-point energy computation at the rhf/cc-pvdz level:

```
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
```
$set_molecule

H 0.00000 -0.75906 0.52195
H 0.00000 0.75906 0.52195
$end_molecule

$set Globals

basis cc-pvdz
aux_basis_scf cc-pvdz-jkfit
method scf
reference rhf
jobtype energy
$end Globals

*Single-point energy computation for a molecule in an electric field.

$set_molecule H20

0 1
0 0.00000 0.00000 -0.06577
H 0.00000 -0.75906 0.52195
H 0.00000 0.75906 0.52195
$end_molecule

$set Globals

basis cc-pvdz
aux_basis_scf cc-pvdz-jkfit
method scf
reference rhf
jobtype energy
electric_field xyz
electric_strength [0.1;0.1;0.1]
$end Globals

8.4 SCF Module Options

- **ELECTRIC_FIELD**
  Applies an electric field in the given direction. The electric field strength, which will be applied, is controlled by the EF_STRENGTH option.
  TYPE: String
  DEFAULT: NONE
  OPTIONS: NONE, X, Y, Z, XY, XZ, YZ, XYZ
  - NONE: It does not apply any electric field.
  - X: Applies an electric field in the \( x \)-direction.
  - Y: Applies an electric field in the \( y \)-direction.
  - Z: Applies an electric field in the \( z \)-direction.
- XY: Applies an electric field in the $x$ and $y$ directions.
- XZ: Applies an electric field in the $x$ and $z$ directions.
- YZ: Applies an electric field in the $y$ and $z$ directions.
- XYZ: Applies an electric field in the $x$, $y$, and $z$ directions.

**EF_STRENGTH**

The strength of the electric field applied in the $x$, $y$, and $z$ directions, respectively.

**TYPE:** Double Array  
**DEFAULT:** [0.1; 0.1; 0.1]  
**OPTIONS:** $n$ User defined value.

**DIIS_DAMP**

Scale diagonal elements of matrix $B$ by $1 + d$. It helps to speed up SCF converge in problematic cases, especially in open-shell systems. The damping parameter is controlled by the DAMP_PARAM option.

**TYPE:** Boolean  
**DEFAULT:** FALSE  
**OPTIONS:** TRUE, FALSE

**DAMP_PARAM**

The damping parameter $d$.

**TYPE:** Double  
**DEFAULT:** 0.02  
**OPTIONS:** $n$ User defined value.

**DIE_IF_NOT_CONVERGED**

Die if SCF is not converged?

**TYPE:** Boolean  
**DEFAULT:** FALSE  
**OPTIONS:** TRUE, FALSE

**DIIS**

Do apply DIIS for SCF iterations?

**TYPE:** Boolean  
**DEFAULT:** TRUE  
**OPTIONS:** TRUE, FALSE

**FOCK_DAMP**

Do apply Fock damping during SCF iterations? The extent of damping is controlled by the **FOCK_DAMP_PERC** option.

**TYPE:** Boolean  
**DEFAULT:** FALSE
• **FOCK_DAMP_PERC**  
The percentage of damping to apply to the early Fock updates. 100 will completely stall the update, while 0 will result in a full update. A value of 20, which corresponds to 20% of the previous iteration’s Fock matrix being mixed into the current Fock. Fock damping may help to solve problems with oscillatory convergence.  
TYPE: Double  
DEFAULT: 20.0  
OPTIONS: 0-100

• **GUESS_MIX**  
Do mix HOMO and LUMO? The GUESS_MIX option may be used for singlet molecules when UHF reference is employed.  
TYPE: Boolean  
DEFAULT: FALSE  
OPTIONS: TRUE, FALSE

• **GUESS_PREVIOUS**  
Do use orbitals of previous geometry? This option is used for geometry optimization, where SCF orbitals guess is obtained from the previous geometry.  
TYPE: Boolean  
DEFAULT: TRUE  
OPTIONS: TRUE, FALSE

• **LOCAL**  
Do localize occupied molecular orbitals after SCF convergence?  
TYPE: Boolean  
DEFAULT: FALSE  
OPTIONS: TRUE, FALSE

• **SAD_FRAC_OCC**  
Do force an even distribution of occupations across the last partially occupied orbital shell?  
TYPE: Boolean  
DEFAULT: TRUE  
OPTIONS: TRUE, FALSE

• **SAD_SPIN_AVERAGE**  
Do use spin-averaged occupations instead of atomic ground spin state in fractional SAD?  
TYPE: Boolean
• SAD_DIIIS
  Do apply DIIS for atomic UHF density computation iterations in SAD?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

• CUTOFF
  The tolerance value used for numerical procedures.
  TYPE: Double
  DEFAULT: 1.0e-10
  OPTIONS: n User defined tolerance value.

• E_CONVERGENCE
  Convergence criterion for the energy.
  TYPE: Double
  DEFAULT: 1.0e-8
  OPTIONS: n User-defined tolerance value.

• D_CONVERGENCE
  Convergence criterion for the density matrix.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined tolerance value.

• QC_GUESS_E_CONVERGENCE
  Convergence criterion for the energy for the QC-SCF guess.
  TYPE: Double
  DEFAULT: 1.0e-4
  OPTIONS: n User-defined tolerance value.

• QC_GUESS_D_CONVERGENCE
  Convergence criterion for the density matrix the QC-SCF guess.
  TYPE: Double
  DEFAULT: 5.0e-3
  OPTIONS: n User-defined tolerance value.

• MIN_OVERLAP_EIGVAL
  Threshold value to eliminate low eigenvalues of the overlap matrix.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined tolerance value.
• **LOCAL CONVERGENCE**
  Convergence criterion used in orbital localization procedure.
  TYPE: Double
  DEFAULT: 1.0e-8
  OPTIONS: n User-defined tolerance value.

• **SAD_CHOLESKY_TOLERANCE**
  SAD guess density decomposition threshold.
  TYPE: Double
  DEFAULT: 1.0e-7
  OPTIONS: n User-defined tolerance value.

• **SAD_E_CONVERGENCE**
  Convergence criterion for the energy in SAD guess.
  TYPE: Double
  DEFAULT: 1.0e-5
  OPTIONS: n User-defined tolerance value.

• **SAD_D_CONVERGENCE**
  Convergence criterion for the density matrix in SAD guess.
  TYPE: Double
  DEFAULT: 1.0e-5
  OPTIONS: n User-defined tolerance value.

• **MAX_DIIS**
  Maximum number of vectors used in the DIIS procedure.
  TYPE: Integer
  DEFAULT: 6
  OPTIONS: n User-defined value.

• **MIN_DIIS**
  Minimum number of vectors used in the DIIS procedure.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: n User-defined value.

• **PRINT_LEVEL**
  Controls the amount of extra information to be printed.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.

• **SCF_MAXITER**
  Maximum number of iterations in the SCF procedure.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: n User-defined value.
• **LOCAL_MAXITER**
  Maximum number of iterations in the localization procedure.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: $n$ User-defined value.

• **SAD_MAXITER**
  Maximum number of iterations in the SAD guess.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: $n$ User-defined value.

• **SAD_MAX_DIIS**
  Maximum number of vectors used in the DIIS procedure in SAD guess.
  TYPE: Integer
  DEFAULT: 6
  OPTIONS: $n$ User-defined value.

• **SAD_MIN_DIIS**
  Minimum number of vectors used in the DIIS procedure in SAD guess.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: $n$ User-defined value.

• **SAD_PRINT**
  Controls the amount of extra information to be printed in SAD guess.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: $n$ User-defined value.

• **DIIS_ALGORITHM**
  The algorithm to store DIIS intermediates.
  TYPE: String
  DEFAULT: DISK
  OPTIONS: DISK, MEM
    - DISK: Store intermediates in DISK.
    - MEM: Store intermediates in the core memory.

• **GUESS**
  Initial orbital guess for the SCF procedure.
  TYPE: String
  DEFAULT: GWH
  OPTIONS: GWH, SAD, CORE
    - GWH: Generalized Wolfsberg-Helmholtz guess.
    - SAD: Superposition of atomic density.
- **CORE**: Core Hamiltonian guess.

- **ORTH_TYPE**
  The orthogonalization algorithm used for the overlap matrix.
  TYPE: String
  DEFAULT: CANONICAL
  OPTIONS: CANONICAL, SYMMETRIC
  - CANONICAL: Canonical orthogonalization.
  - SYMMETRIC: Symmetric orthogonalization.

- **ERI**
  Controls the handling of electron repulsion integrals (ERIs).
  TYPE: String
  DEFAULT: INCORE
  OPTIONS: INCORE, DIRECT
  - INCORE: Incore algorithm to build the Fock matrix.
  - DIRECT: Integral direct/semi-direct algorithm to build the Fock matrix.

- **SCF_ALGORITHM**
  Controls the SCF algorithm.
  TYPE: String
  DEFAULT: ROOTHAAN
  OPTIONS: ROOTHAAN, ASO, QC
  - ROOTHAAN: Roothaan algorithm.
  - ASO: Approximately second-order SCF.
  - QC: Quadratically convergent SCF.

- **LOCAL_TYPE**
  Controls the orbital localization algorithm.
  TYPE: String
  DEFAULT: BOYS
  OPTIONS: BOYS, PIPEK_MEZEY

- **ROHF_CANONIC_TYPE**
  Controls the orbital localization algorithm.
  TYPE: String
  DEFAULT: GUEST
  OPTIONS: GUEST, ROOTHAAN, MCWEENY, GAMESS, DAVIDSON, BINKLEY, FAEGRI, PLAKHUTIN, ALPHA, BETA
9  OPT : Geometry Optimization

Code Authors: U. Bozkaya and B. Ermiş.

9.1  Overview

MacroQC performs geometry optimizations (minimization and first-order saddle points, the latter one is experimental at this stage) for a variety of molecular structures using either analytic or numerical energy gradients. MacroQC has a rich analytic gradient package; hence, in most cases, one can take advantage of analytic gradients. The present version of MacroQC uses Cartesian coordinates in geometry optimizations. MacroQC has several optimization algorithms, such as steepest-descent, quasi-Newton, and rational-function optimization algorithms [37] [38] [39].

9.2  Input Examples

*Geometry optimization at the rhf/cc-pvdz level:

```bash
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set globals
  basis cc-pvdz
  aux_base_scf cc-pvdz-jkfit
  method scf
  reference rhf
  jobtype opt
$end globals

$set_opt
  opt_algorithm qn
  opt_maxiter 100
  tol_rmsG 3.0E-4
  tol_maxG 4.5E-4
  tol_rmsDisp 1.2E-3
  tol_maxDisp 1.8E-3
$end_opt

*Starting geometry optimization by reading the Hessian. :
```

In order to do this, there must be an `input_prefix.hess` file. (FREQ module generates
9.3 OPT Module Options

- **OPT_TYPE**
  Type of the optimization.
  TYPE: String  
  DEFAULT: MIN  
  OPTIONS: MIN, TS
  
  - MIN : Minimum search
  - TS : Transition-state search

- **OPT_ALGORITHM**
  The geometry optimization algorithm.
  TYPE: String  
  DEFAULT: QN  
  OPTIONS: QN, RFO, SD
  
  - QN : Quasi-Newton
- RFO: Rational Function Optimization
- SD: Steepest Descent

**HESS_UPDATE**
The hessian update algorithm.
TYPE: String
DEFAULT: BFGS
OPTIONS: BFGS, BOFILL
  - BFGS: Broyden-Fletcher-Goldfarb-Shanno Hessian update algorithm.
  - BOFILL: BOFILL Hessian update algorithm.

**PRINT_LEVEL**
The amount of information to print to the output file.
TYPE: Integer
DEFAULT: 0
OPTIONS: n User-defined value.

**OPT_MAXITER**
The maximum number of geometry optimization steps.
TYPE: Integer
DEFAULT: 100
OPTIONS: n User-defined value.

**FOLLOW_ROOT**
Root for Partial RFO to follow.
TYPE: Integer
DEFAULT: 1
OPTIONS: n User-defined value.

**ALPHA**
The line search parameter.
TYPE: Double
DEFAULT: 1.0
OPTIONS: n User-defined value.

**MAX_STEP**
Maximum geometry optimization step size in Bohr.
TYPE: Double
DEFAULT: 0.5
OPTIONS: n User-defined value.

**LEVEL_SHIFT_PARAM**
Level shift to aid convergence.
TYPE: Double
DEFAULT: 0.01
OPTIONS: n User-defined value.
• **CUTOFF**
The cutoff value for numerical procedures.
TYPE: Double
DEFAULT: 1.0e-10
OPTIONS: $n$ User-defined value.

• **TOL_MAXDISP**
Convergence criterion for geometry optimization: maximum displacement (cartesian coordinates, atomic units).
TYPE: Double
DEFAULT: 1.8e-3
OPTIONS: $n$ User-defined value.

• **TOL_RMSDISP**
Convergence criterion for geometry optimization: rms displacement (cartesian coordinates, atomic units).
TYPE: Double
DEFAULT: 1.2e-3
OPTIONS: $n$ User-defined value.

• **TOL_MAXG**
Convergence criterion for geometry optimization: maximum gradient (cartesian coordinates, atomic units).
TYPE: Double
DEFAULT: 4.5e-4
OPTIONS: $n$ User-defined value.

• **TOL_RMSG**
Convergence criterion for geometry optimization: rms gradient (cartesian coordinates, atomic units).
TYPE: Double
DEFAULT: 3.0e-4
OPTIONS: $n$ User-defined value.

• **DIATOMIC_OPTIMIZER**
MacroQC includes a separate optimizer for diatomic molecules. This optimizer is automatically selected for diatomic molecules. The diatomic optimizer generates 5 displaced geometries around the initial geometry and computes energy at each perturbed geometry. Then, using a 5-point interpolation formula, an analytic potential energy function is generated. Using this analytic potential energy function, geometry optimization, harmonic and anharmonic vibrational frequency computations are performed.
TYPE: Boolean
DEFAULT: TRUE
OPTIONS: TRUE, FALSE

• **WRITE_CART_HESS**
Do write Cartesian Hessian?
• **READ_CART_HESS**
  Do read Cartesian Hessian?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **PROJECT_OUT_RC**
  Do project out redundant coordinate components from Cartesian Hessian?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE
10 FREQ: Vibrational Frequencies

Code Authors: B. Ermiş and U. Bozkaya.

10.1 Overview

The freq module generates displaced geometries from the initial molecular geometry to calculate the harmonic and anharmonic vibrational frequencies of chemical structures [40]. By default, the freq module works in serial mode. However, the parallel mode is also available and recommended for high-cost computations. Each displaced geometry is formed by the infrared program to be a separate input file. The input file type is set according to the selected program (e.g. software program_name). The generated input files are run in the selected program for the gradient calculations. Because these calculations are independent of each other, they can be run simultaneously on different computers. Thus, freq enables one to compute infrared spectroscopy at a lower cost compared with the serial mode.

The package comprises:

- Displaced geometry
- Harmonic vibrational frequencies
- Anharmonic vibrational frequencies
- Thermodynamic analysis
- Infrared intensity

- The freq module creates epsilon_harm.out or epsilon_anharm.out file to visualize the infrared spectrum. It also creates a file in molden format to animate vibrational modes.

The parallel mode of the freq module can be integrated with several quantum chemistry software such as Psi4 [11], Qchem [42], Cfour [44], Molpro [45], and Orca [46].

10.2 Input Examples

*Harmonic vibrational frequency computation at the mp2/cc-pvdz level in serial mode:

```
$set_molecule H2O
  0 1
  O  0.00000  0.00000 -0.06577
  H  0.00000 -0.75906  0.52195
  H  0.00000  0.75906  0.52195
$end_molecule
```
Anharmonic vibrational frequency computation at the mp2/cc-pvdz level in parallel mode is given below. By selecting MODE SOW, a separate file is created for each gradient computation. Make sure the method you choose has an analytic gradient.
When each gradient computation is completed, you can proceed to the frequency computation by selecting MODE REAP:

```
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set_globals
  jobtype freq
$end_globals

$set_freq
  step_bohr 0.05
  anharm_level 2
  derive_level 4
  mode reap
  temperature 298.15
  pressure 101325.0
  energy -76.22842455
$end_freq
```

*Anharmonic vibrational frequency computation at the mp2/cc-pvdz level in parallel mode is given below. In this example the PSI4 software is employed instead of MacroQC. When an external software is used, the options for the external software should be provided in the DG_EXTERNAL block. MacroQC generates all necessary input files for the allowed external software. By selecting MODE SOW, a separate file is created for each gradient computation. Make sure the method you choose has an analytic gradient. When each gradient computation is completed, you can proceed to the frequency computation by selecting MODE REAP:

```
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set_globals
  jobtype freq
$end_globals
```
10.3 FREQ Module Options

- **MODE**
  Type of the mode.
  TYPE: String
  DEFAULT: SERIAL
  OPTIONS: SERIAL, SOW, REAP
  
  - SERIAL: It allows the frequency computation to run in the serial mode.
  
  - SOW: Creates perturbed geometries and allows you to run them in parallel by generating input files for displaced geometries.
  
  - REAP: When analytic gradient computations are completed for displaced geometries, REAP option requests to compute vibrational frequencies. Note that when the MacroQC program is executed with REAP option, all outputs for displaced geometries should be available in the working directory.

- **SOFTWARE**
  Which quantum chemistry software to use?
  TYPE: String
DEFAULT: MACROQC
OPTIONS: MACROQC, PSI4, QCHEM, MOLPRO, ORCA, CFOUR

- **DERIVE_LEVEL**
  Specifies the level of energy derivatives.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: 2, 3, 4

- **ANHARM_LEVEL**
  Do compute anharmonic frequencies?
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: 0, 1, 2
  - 0: Harmonic vibration frequency.
  - 1: Anharmonic correction (cubic terms).
  - 2: Anharmonic correction (quartic terms).

- **PRINT_LEVEL**
  The amount of information to print to the output file.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: $n$ User-defined value.

- **STEP_BOHR**
  Specifies the step-size for numerical differentiation (in atomic units). The same value should be used for the SOW and REAP options.
  TYPE: Double
  DEFAULT: 0.05
  OPTIONS: $n$ User-defined value.

- **TOL_ANHARM**
  Tolerance value for Fermi resonance [47].
  TYPE: Double
  DEFAULT: 200.0
  OPTIONS: $n$ User-defined value.

- **TOL_MARTIN**
  Tolerance value for criterion suggested by Martin and co-workers [48].
  TYPE: Double
  DEFAULT: 10.0
  OPTIONS: $n$ User-defined value.

- **W_LORENTZ**
  Lorentz width at half height.
  TYPE: Double
DEFAULT: 10.0
OPTIONS: n User-defined value.

- **PRESSURE**
  Pressure (in Pascal).
  TYPE: Double
  DEFAULT: 101325.0
  OPTIONS: n User-defined value.

- **TEMPERATURE**
  Temperature (in Kelvin).
  TYPE: Double
  DEFAULT: 298.15
  OPTIONS: n User-defined value.

- **ENERGY**
  Electronic energy at the optimized geometry (in atomic units).
  TYPE: Double
  DEFAULT: 298.15
  OPTIONS: n User-defined value.

- **POINT_GROUP**
  Full point group.
  TYPE: String
  DEFAULT: C1
  OPTIONS: ATOM, C1, CI, CS, C_INF_V, D_INF_H, TD, OH, IH, CN, CNV, CNH, DN, DND, DNH, SN

- **AXIS_N**
  Point group axis, which is needed to determine rotational symmetry number. Users need to provide this option in the case of CN, CNV, CNH, DN, DND, DNH, and SN point groups. In the case of other point groups, it is unnecessary to set this option.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: n User-defined value. For CN, CNV, CNH, DN, DND, DNH, and SN point groups n are equal to N of the point group. For example for C2V group n = 2.
11  DFOCC: Orbital-Optimized Coupled-Cluster and Møller–Plesset Perturbation Theories

Code Authors: U. Bozkaya with contributions from A. Ünal and Y. Alagöz.

11.1  Theory

What follows is a very basic description of orbital-optimized Møller–Plesset perturbation theory as implemented in Macroqc.

The orbital variations may be expressed by means of an exponential unitary operator

\[
\tilde{\hat{p}}^{\dagger} = e^{\hat{K}} \hat{p}^{\dagger} e^{-\hat{K}} \quad (6)
\]

\[
\tilde{\hat{p}} = e^{\hat{K}} \hat{p} e^{-\hat{K}} \quad (7)
\]

\[
|\tilde{p}\rangle = e^{\hat{K}} |p\rangle \quad (8)
\]

where \( \hat{K} \) is the orbital rotation operator

\[
\hat{K} = \sum_{p,q} K_{pq} \hat{E}_{pq} = \sum_{p>q} \kappa_{pq} \hat{E}^{-}_{pq} \quad (9)
\]

\[
\hat{E}_{pq} = \hat{p}^{\dagger} \hat{q} \quad (10)
\]

\[
\hat{E}^{-}_{pq} = \hat{E}_{pq} - \hat{E}_{qp} \quad (11)
\]

\[
\hat{K} = \text{Skew}(\kappa) \quad (12)
\]

The effect of the orbital rotations on the MO coefficients can be written as

\[
\mathbf{C}(\kappa) = \mathbf{C}^{(0)} e^{\mathbf{K}} \quad (13)
\]

where \( \mathbf{C}^{(0)} \) is the initial MO coefficient matrix and \( \mathbf{C}(\kappa) \) is the new MO coefficient matrix as a function of \( \kappa \). Now, let us define a variational energy functional (Lagrangian) as a function of \( \kappa \),

OMP2 \([54, 57, 20, 19, 21]\):

\[
\tilde{E}(\kappa) = \langle 0 | \hat{H}^{\kappa} | 0 \rangle + \langle 0 | (\hat{W}_{N}^{\kappa} \hat{T}_{2}^{(1)})_{c} | 0 \rangle + \langle 0 | \{ \hat{A}_{2}^{(1)} (\hat{f}_{N}^{\kappa} \hat{T}_{2}^{(1)}) + \hat{W}_{N}^{\kappa} \}_{c} | 0 \rangle, \quad (14)
\]

OMP3 \([55, 65, 59, 22, 27]\):

\[
\tilde{E}(\kappa) = \langle 0 | \hat{H}^{\kappa} | 0 \rangle + \langle 0 | (\hat{W}_{N}^{\kappa} \hat{T}_{2}^{(1)})_{c} | 0 \rangle + \langle 0 | (\hat{W}_{N}^{\kappa} \hat{T}_{2}^{(2)})_{c} | 0 \rangle + \langle 0 | \{ \hat{A}_{2}^{(1)} (\hat{f}_{N}^{\kappa} \hat{T}_{2}^{(1)}) + \hat{W}_{N}^{\kappa} \}_{c} | 0 \rangle + \langle 0 | \{ \hat{A}_{2}^{(1)} (\hat{f}_{N}^{\kappa} \hat{T}_{2}^{(2)}) + \hat{W}_{N}^{\kappa} \}_{c} | 0 \rangle + \langle 0 | \{ \hat{A}_{2}^{(2)} (\hat{f}_{N}^{\kappa} \hat{T}_{2}^{(1)}) + \hat{W}_{N}^{\kappa} \}_{c} | 0 \rangle, \quad (15)
\]

57
OMP2.5 \([66, 67, 22]\):

\[
\tilde{E}(\kappa) = \langle 0 | \hat{H} | 0 \rangle \\
+ \left\langle 0 \left| \left( \hat{W}_{\kappa}^{(1)} \hat{T}_{2}^{(1)} \right)_{c} \right| 0 \right\rangle \\
+ \frac{1}{2} \left\langle 0 \left| \left( \hat{W}_{\kappa}^{(2)} \hat{T}_{2}^{(2)} \right)_{c} \right| 0 \right\rangle \\
+ \frac{1}{2} \left\langle 0 \left| \left\{ \hat{\Lambda}_{2}^{(1)} \left( \hat{f}_{\kappa}^{*} \hat{T}_{2}^{(1)} \hat{W}_{\kappa}^{*} \right)_{c} \right| 0 \right\rangle \\
+ \frac{1}{2} \left\langle 0 \left| \left\{ \hat{\Lambda}_{2}^{(2)} \left( \hat{f}_{\kappa}^{*} \hat{T}_{2}^{(2)} \hat{W}_{\kappa}^{*} \right)_{c} \right| 0 \right\rangle ,
\]

(16)

OLCCD \([58, 68, 23]\):

\[
\tilde{E}(\kappa) = \langle 0 | \hat{H}^{\kappa} | 0 \rangle \\
+ \left\langle 0 \left| \left( \hat{W}_{\kappa}^{*} \hat{T}_{2} \right)_{c} \right| 0 \right\rangle \\
+ \left\langle 0 \left| \left\{ \hat{\Lambda}_{2} \left( \hat{W}_{\kappa}^{*} + \hat{H}_{\kappa}^{*} \hat{T}_{2} \right) \right)_{c} \right| 0 \right\rangle ,
\]

(17)

OCCD \([54, 62]\):

\[
\tilde{E}(\kappa) = \langle 0 | (1 + \hat{\Lambda}_{2}) e^{-\hat{T}_{2}} \hat{H}^{\kappa} e^{\hat{T}_{2}} | 0 \rangle ,
\]

(18)

where subscript \(c\) means only connected diagrams are included, \(\hat{T}_{2}\) and \(\hat{\Lambda}_{2}\) are cluster double excitation and de-excitation operators, respectfuully, and \(\hat{H}^{\kappa}, \hat{f}_{\kappa}^{*}, \hat{W}_{\kappa}^{*}, \) and \(\hat{H}_{\kappa}^{*}\) are defined as

\[
\hat{H}^{\kappa} = e^{-\hat{K}} \hat{H} e^{\hat{K}},
\]

(19)

\[
\hat{f}_{\kappa}^{*} = e^{-\hat{K}} \hat{f}^{*} e^{\hat{K}},
\]

(20)

\[
\hat{W}_{\kappa}^{*} = e^{-\hat{K}} \hat{W} e^{\hat{K}},
\]

(21)

\[
\hat{H}_{\kappa}^{*} = e^{-\hat{K}} \hat{H} e^{\hat{K}}.
\]

(22)

The energy gradient and Hessian can be written as follows:

\[
w_{pq} = \frac{\partial \tilde{E}}{\partial \kappa_{pq}} \bigg|_{\kappa=0},
\]

(23)

\[
A_{pq,rs} = \frac{\partial^{2} \tilde{E}}{\partial \kappa_{pq} \partial \kappa_{rs}} \bigg|_{\kappa=0},
\]

(24)

Then the energy can be expanded up to second-order as follows:

\[
\tilde{E}^{(2)}(\kappa) = \tilde{E}^{(0)} + \kappa^{\dagger} w + \frac{1}{2} \kappa^{\dagger} A \kappa,
\]

(25)

where \(\kappa\) is the MO rotation vector, \(w\) is the MO gradient vector, and \(A\) is the MO Hessian matrix. Hence, minimizing the energy with respect to \(\kappa\) yields

\[
\kappa = -A^{-1} w.
\]

(26)
This final equation corresponds to the usual Newton-Raphson step.

The orbital-optimized MPn and CC methods currently supported in dfocc are outlined in Table 7.

Publications resulting from the use of the orbital-optimized code should cite the following publications:

**OMP2** Bozkaya and co-workers [54, 57, 20, 19, 21].

**OMP3** Bozkaya and co-workers [55, 65, 59, 22, 27].

**OMP2.5** Bozkaya and co-workers [66, 67, 22].

**OLCCD** Bozkaya and co-workers [58, 68, 23].

**OCCD** Bozkaya and co-workers [62].

11.2 Conventional (Non-OO) Coupled-Cluster and Moller-Plesset Perturbation Theories

Non-orbital-optimized counterparts to MPn and CC methods currently supported in dfocc are outlined in Table 8.

Publications resulting from the use of the MPn and CC codes should cite the following publications:

Table 7: OO methods.

<table>
<thead>
<tr>
<th>Name</th>
<th>Calls Method</th>
<th>Energy</th>
<th>Gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>omp2</td>
<td>Density-Fitted Orbital-Optimized MP2</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>omp3</td>
<td>Density-Fitted Orbital-Optimized MP3</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>omp2.5</td>
<td>Density-Fitted Orbital-Optimized MP2.5</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>olccd</td>
<td>Density-Fitted Orbital-Optimized LCCD</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>occd</td>
<td>Density-Fitted Orbital-Optimized CCD</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>occd(t)</td>
<td>Density-Fitted Orbital-Optimized CCD(T)</td>
<td>RHF/UHF</td>
<td>NA</td>
</tr>
<tr>
<td>occd(at)</td>
<td>Density-Fitted Orbital-Optimized CCD(AT)</td>
<td>RHF/UHF</td>
<td>NA</td>
</tr>
</tbody>
</table>

Table 8: Non-OO methods.

<table>
<thead>
<tr>
<th>Name</th>
<th>Calls Method</th>
<th>Energy</th>
<th>Gradient</th>
</tr>
</thead>
<tbody>
<tr>
<td>mp2</td>
<td>Density-Fitted MP2</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>mp3</td>
<td>Density-Fitted MP3</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>mp2.5</td>
<td>Density-Fitted MP2.5</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>lccd</td>
<td>Density-Fitted LCCD</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>ccd</td>
<td>Density-Fitted CCD</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>ccisd</td>
<td>Density-Fitted CCSD</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>ccisd(t)</td>
<td>Density-Fitted CCSD(T)</td>
<td>RHF/UHF</td>
<td>RHF/UHF</td>
</tr>
<tr>
<td>ccisd(at)</td>
<td>Density-Fitted CCSD(AT)</td>
<td>RHF/UHF</td>
<td>NA</td>
</tr>
</tbody>
</table>

59
**MP2** Bozkaya and co-workers [54, 57, 20, 19, 21].
**MP3** Bozkaya and co-workers [55, 65, 59, 22, 27].
**MP2.5** Bozkaya and co-workers [66, 67, 22].
**LCCD** Bozkaya and co-workers [58, 68, 23].
**LCCD** Bozkaya and co-workers [58, 68, 23].
**CCD** Bozkaya and co-workers [54, 24, 25, 62].
**CCSD** Bozkaya and co-workers [54, 24, 25, 62].
**CCSD(T)** Bozkaya and co-workers [56, 24, 26].
**CCSD(AT)** Bozkaya and co-workers [56, 24].

11.3 Input Examples

*Single point omp2/cc-pvdz computation:

```bash
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basisscfc cc-pvdz-jkfit
  aux_basiscorr cc-pvdz-ri
  method omp2
  reference rhf
  jobtype energy
  freeze_core true
$end_globals

$set_dfocc
  e_convergence 1e-8
  r_convergence 1e-6
  mo_maxiter 50
  max_mograd_convergence 1.0e-3
  rms_mograd_convergence 1.0e-6
$end_dfocc
```

*Single point ccsd(t)/cc-pvdz computation:

```bash
$set_molecule H2O
  0 1
```

60
*Single point fno-ccsd(t)/cc-pvdz computation:*

```plaintext
$set_molecule H2O
  0 1
  0 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set Globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method ccsd(t)
  reference rhf
  jobtype energy
  freeze_core true
$end_globals

$set_dfocc
  e_convergence 1e-8
  r_convergence 1e-6
$end_dfocc
```
*Ionization potentials via the extended Koopmans’ theorem (EKT) at the ccsd(t)/cc-pvdz level:

```plaintext
$set_molecule H2O
  0 1
  O 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method ccsd(t)
  reference rhf
  jobtype grad
  freeze_core true
$end_globals

$set_dfocc
  e_convergence 1e-8
  r_convergence 1e-6
  ekt_ip true
$end_dfocc
```

*Frozen-natural orbitals at the ccsd(t)/cc-pvdz level:

```plaintext
$set_molecule H2O
  0 1
  O 0.00000 0.00000 -0.06577
  H 0.00000 -0.75906 0.52195
  H 0.00000 0.75906 0.52195
$end_molecule

$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method ccsd(t)
  reference rhf
  jobtype energy
  freeze_core true
$end_globals

$set_dfocc
```
11.4 DFOCC Module Options

- **DAVIDSON_TOLERANCE**
  Convergence criterion for Davidson (residuals).
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined tolerance value.

- **E3_SCALE**
  TYPE: Double
  DEFAULT: 0.25
  OPTIONS: n User-defined value.

- **E_CONVERGENCE**
  Convergence criterion for energy.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined value.

- **LEVEL_SHIFT**
  Level shift to aid convergence.
  TYPE: Double
  DEFAULT: 0.02
  OPTIONS: n User-defined value.

- **MAX_MOGRAD_CONVERGENCE**
  Convergence criterion for maximum orbital gradient.
  TYPE: Double
  DEFAULT: 1.0e-3
  OPTIONS: n User-defined value.

- **MO_STEP_MAX**
  Maximum step size in orbital-optimization procedure.
  TYPE: Double
  DEFAULT: 0.5
  OPTIONS: n User-defined value.
• **MP2_OS_SCALE**  
  MP2 opposite-spin scaling value.  
  TYPE: Double  
  DEFAULT: $6.0/5.0$  
  OPTIONS: $n$ User-defined value.

• **MP2_SOS_SCALE**  
  MP2 Spin-opposite scaling (SOS) value.  
  TYPE: Double  
  DEFAULT: $1.3$  
  OPTIONS: $n$ User-defined value.

• **MP2_SOS_SCALE2**  
  Spin-opposite scaling (SOS) value for optimized-MP2 orbitals.  
  TYPE: Double  
  DEFAULT: $1.2$  
  OPTIONS: $n$ User-defined value.

• **MP2_SS_SCALE**  
  MP2 same-spin scaling value.  
  TYPE: Double  
  DEFAULT: $1.0/3.0$  
  OPTIONS: $n$ User-defined value.

• **OCC_PERCENTAGE**  
  Cutoff for the occupation of MP2 virtual NOs in FNO-CCSD/CCSD(T). The number of virtual NOs is chosen so the occupation of the truncated virtual space is OCC_PERCENTAGE percent of occupation of the original MP2 virtual space. This option is only used if NAT_ORBS = true. This keyword overrides OCC_TOLERANCE.  
  TYPE: Double  
  DEFAULT: $99.0$  
  OPTIONS: $n$ User-defined value.

• **OCC_TOLERANCE**  
  Cutoff for the occupation of MP2 virtual NOs in FNO-CCSD/CCSD(T). Virtual NOs with occupations less than OCC_TOLERANCE will be discarded. This option is only used if NAT_ORBS = true.  
  TYPE: Double  
  DEFAULT: $1.0e-5$  
  OPTIONS: $n$ User-defined tolerance value.

• **OO_SCALE**  
  OO scaling factor used in MSD.  
  TYPE: Double  
  DEFAULT: $0.01$  
  OPTIONS: $n$ User-defined value.
- **PCG_CONVERGENCE**
  Convergence criterion for the residual vector of the preconditioned conjugate gradient method.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: \( n \) User-defined value.

- **SCG_TOLERANCE**
  Tolerance for strictly canonical.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: \( n \) User-defined tolerance value.

- **R_CONVERGENCE**
  Convergence criterion for amplitudes (residuals).
  TYPE: Double
  DEFAULT: 1.0e-5
  OPTIONS: \( n \) User-defined value.

- **REG_PARAM**
  Regularization parameter.
  TYPE: Double
  DEFAULT: 0.4
  OPTIONS: \( n \) User-defined value.

- **RMS_MOGRAD_CONVERGENCE**
  Convergence criterion for RMS orbital gradient. Default adjusts depending on E_CONVERGENCE.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: \( n \) User-defined value.

- **ACTIVE_NAT_ORBS**
  An array containing the number of virtual natural orbitals per irrep (in Cotton order) so a user can specify the number of retained natural orbitals rather than determining them with OCC_TOLERANCE. This keyword overrides OCC_TOLERANCE and OCC_PERCENTAGE.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: \( n \) User-defined value.

- **CC_DIIS_MAX_VECS**
  Maximum number of vectors used in amplitude DIIS.
  TYPE: Integer
  DEFAULT: 6
  OPTIONS: \( n \) User-defined value.
• **CC\_DIIS\_MIN\_VECS**
  Minimum number of vectors used in amplitude DIIS.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: \(n\) User-defined value.

• **CC\_MAXITER**
  The maximum number of iterations to determine the amplitudes. This option is also used for CIS iterations.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: \(n\) User-defined value.

• **CUTOFF**
  The cutoff value for numerical procedures.
  TYPE: Integer
  DEFAULT: 8
  OPTIONS: \(n\) User-defined value.

• **FOLLOW\_ROOT**
  Follow QDPT root.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: \(n\) User-defined value.

• **INTEGRAL\_CUTOFF**
  Cutoff value for DF integrals.
  TYPE: Integer
  DEFAULT: 9
  OPTIONS: \(n\) User-defined value.

• **MO\_DIIS\_NUM\_VECS**
  The number of vectors used in orbital DIIS.
  TYPE: Integer
  DEFAULT: 6
  OPTIONS: \(n\) User-defined value.

• **MO\_MAXITER**
  Maximum number of iterations to determine the orbitals.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: \(n\) User-defined value.

• **NUM\_ROOTS**
  Number of CI Roots interested.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: \(n\) User-defined value.
• **PCG_MAXITER**  
  Maximum number of preconditioned conjugate gradient iterations.  
  TYPE: Integer  
  DEFAULT: 50  
  OPTIONS: $n$ User-defined value.

• **PRINT_LEVEL**  
  The amount of information to print to the output file.  
  TYPE: Integer  
  DEFAULT: 0  
  OPTIONS: $n$ User-defined value.

• **CC_LAMBDA**  
  Do solve lambda amplitude equations?  
  TYPE: Boolean  
  DEFAULT: FALSE  
  OPTIONS: TRUE, FALSE.

• **CIS_ALGORITHM**  
  CIS algorithm.  
  TYPE: String  
  DEFAULT: MO_BASIS  
  OPTIONS: MO_BASIS

• **CIS_GUESS_COEFF**  
  Initial guess for unrestricted CIS (UCIS) computations.  
  TYPE: String  
  DEFAULT: MIX  
  OPTIONS: MIX, SINGLET, TRIPLET

  – MIX: Use a mixture of singlet and triplet states.
  – SINGLET: Use singlet coefficients, which means $c_i^A = 1/\sqrt{2}$ and $c_i^a = 1/\sqrt{2}$.
  – TRIPLET: Use triplet coefficients, which means $c_i^A = 1/\sqrt{2}$ and $c_i^a = -1/\sqrt{2}$.

• **UNIT_GUESS_TYPE**  
  Type of initial guesses.  
  TYPE: String  
  DEFAULT: GEN  
  OPTIONS: LIMITED, GEN

  – LIMITED: This option requests a limited guess to be generated around HOMO-LUMO orbitals. The dimension of guess is determined by the EXCITATION_RANGE option.
  – GEN: All singly excited Slater determinants are used in generating an initial guess.
• **EXCITATION_RANGE**
The gap between the occupied and virtual spaces from which initially excited state
guesses are directed to the Davidson algorithm in CIS computations. The default
value of 2 corresponds to HOMO-1, HOMO, LUMO, and LUMO+1.
TYPE: Integer
DEFAULT: 2
OPTIONS: n User-defined value.

• **VECS_PER_ROOT**
Subspace vectors used per root in CIS computations.
TYPE: Integer
DEFAULT: 1
OPTIONS: n User-defined value.

• **SCHMIDT_ADD_RES_TOL**
Tolerance value to add a guess vector of a root to the Davidson algorithm in CIS
computations.
TYPE: Double
DEFAULT: 1.0e-3
OPTIONS: n User-defined tolerance value.

• **CIS_E_CONV**
Convergence criterion for excitation energy.
TYPE: Double
DEFAULT: 1.0e-6
OPTIONS: n User-defined tolerance value.

• **CIS_RES_PRECONDITION_TOL**
Tolerance value of CIS residual preconditioning.
TYPE: Double
DEFAULT: 1.0e-4
OPTIONS: n User-defined tolerance value.

• **MAX_VECS**
Maximum dimension of the subspace.
TYPE: Integer
DEFAULT: 50
OPTIONS: n User-defined value.

• **COMPUT_S2**
Do compute \( \langle S^2 \rangle \) for DF-OMP2/DF-MP2?
TYPE: Boolean
DEFAULT: FALSE
OPTIONS: TRUE, FALSE.

• **DIAG_METHOD**
Diagonalization method used for the CIS Hamiltonian.
- **DAVIDSON**: Use the Davidson algorithm, which is available for RHF and UHF references.
- **FULL_DIAG**: Build the overall CIS Hamiltonian and diagonalize it. This option is available only for the RHF reference.

- **DO_DIIS**
  Do apply DIIS extrapolation?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **DO_LEVEL_SHIFT**
  Do apply level shifting?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **DO_SCS**
  Do perform spin-component-scaled OMP2 (SCS-OMP2)? In all computations, SCS-OMP2 energy is computed automatically. However, in order to perform geometry optimizations and frequency computations with SCS-OMP2, one needs to set 'DO_SCS' to true.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **DO_SOS???</p>
  Do perform spin-opposite-scaled OMP2 (SOS-OMP2)? In all computations, SOS-OMP2 energy is computed automatically. However, in order to perform geometry optimizations and frequency computations with SOS-OMP2, one needs to set 'DO_SOS' to true.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **EKT_IP**
  Do compute ionization potentials based on the extended Koopmans’ theorem?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **MO_READ**
  Do read MO coefficients from external files?
- **MO_WRITE**
  Do write MO coefficients to external files?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **HESS_TYPE**
  Type of the MO Hessian matrix.
  TYPE: String
  DEFAULT: HF
  OPTIONS: HF, APPROX_DIAG, APPROX_DIAG_EKT, APPROX_DIAG_HF

- **LINEQ_SOLVER**
  The solver will be used for simultaneous linear equations.
  TYPE: String
  DEFAULT: CDGESV
  OPTIONS: CDGESV, FLIN, POLE

- **MP2_AMP_TYPE**
  The algorithm that is used to handle mp2 amplitudes. The DIRECT option means compute amplitudes on the fly whenever they are necessary.
  TYPE: String
  DEFAULT: DIRECT
  OPTIONS: DIRECT, CONV

- **NAT_ORBS**
  Do use MP2 NOs to truncate virtual space for CCD/CCSD and (T)?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **NBO**
  Do compute natural orbitals?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **OCC_ORBS_PRINT**
  Do print OCC orbital energies?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE
• **OEPROP**
  Do compute one electron properties?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **OPT_METHOD**
  The orbital optimization algorithm. Presently quasi-Newton-Raphson algorithm is available with several Hessian options.
  TYPE: String
  DEFAULT: QNR
  OPTIONS: QNR

• **ORB_RESP_SOLVER**
  The algorithm will be used for solving the orbital-response equations. The LINEQ option creates the MO Hessian and solves the simultaneous linear equations with the method chosen by the LINEQ_SOLVER option. The PCG option does not create the MO Hessian explicitly, instead, it solves the simultaneous equations iteratively with the preconditioned conjugate gradient method.
  TYPE: String
  DEFAULT: PCG
  OPTIONS: PCG, LINEQ

• **ORTH_TYPE**
  The algorithm for orthogonalization of MOs
  TYPE: String
  DEFAULT: MGS
  OPTIONS: MGS, GS

• **PCG_BETA_TYPE**
  Type of PCG beta parameter (Fletcher-Reeves or Polak-Ribiere).
  TYPE: String
  DEFAULT: FLETCHER_REEVES
  OPTIONS: FLETCHER_REEVES, POLAK_RIBIERE

• **PPL_TYPE**
  Type of the CCSD PPL term.
  TYPE: String
  DEFAULT: AUTO
  OPTIONS: AUTO, LOW_MEM, HIGH_MEM, CD

• **PRINT_CI_VECS**
  Do print CI coefficients in CIS computations?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE
- **QCHF**
  Do perform a QCHF computation?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **REGULARIZATION**
  Do use regularized denominators?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **SCS_TYPE**
  Type of the SCS method.
  TYPE: String
  DEFAULT: SCS
  OPTIONS: SCS, SCSN, SCSVDW, SCSMI

- **SOS_TYPE**
  Type of the SOS method.
  TYPE: String
  DEFAULT: SOS
  OPTIONS: SOS, SOSPI

- **TRIPLES_IABC_TYPE**
  The algorithm to handle (ia|bc) type integrals that are used for (T) correction.
  TYPE: String
  DEFAULT: DISK
  OPTIONS: DISK, INCORE, AUTO, DIRECT

- **LOW_MEM_MP2**
  Do low mem mp2.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **EOM**
  Do compute eom intermediates for EOM-CCSD computations?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **EKT_NEGATIVE_OCC**
  The algorithm to eliminate negative occupation numbers for density matrix in EKT computations.
  TYPE: String
  DEFAULT: ABS
  OPTIONS: ABS, SPN
– ABS: Takes absolute values of the negative occupation numbers.
– SPN: Replace the negative occupation numbers with a smallest positive number, such as 1.0E-15.

12 QDPT: Quasidegenerate Perturbation Theory

Code Author: U. Bozkaya. The qdpt module includes CAS-CI, QDPT2, and FCI methods. Publications resulting from the use of qdpt module should cite Bozkaya [28].

12.1 Input Examples

*Single point qdpt2/cc-pvdz computation with cms(2,1,1), which corresponds to cas(2,2) active space:

```
$set_molecule H2O
  0 1
  O  0.00000  0.00000  -0.06577
  H  0.00000  -0.75906  0.52195
  H  0.00000   0.75906  0.52195
$end_molecule

$setGlobals
  basis cc-pvdz
  method qdpt2
  reference uhf
  jobtype energy
  freeze_core false
$endGlobals

$qdpt
  qdpt_maxiter 50
  nactmo 2
  nalp 1
  nbeta 1
$end_qdpt
```

*Single point cas-ci/cc-pvdz computation with cms(2,1,1), which corresponds to cas(2,2) active space:

```
$set_molecule H2O
  0 1
  O  0.00000  0.00000  -0.06577
  H  0.00000  -0.75906  0.52195
  H  0.00000   0.75906  0.52195
```
12.2 QDPT Module Options

- DAVIDSON_TOLERANCE
  Convergence criterion for Davidson (residuals).
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined tolerance value.
• **E_CONVERGENCE**
  Convergence criterion for energy.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: $n$ User-defined value.

• **ON_MAX**
  Maximum value for occupation number cutoff.
  TYPE: Double
  DEFAULT: 1.98
  OPTIONS: $n$ User-defined value.

• **ON_MIN**
  The minimum value for occupation number cutoff.
  TYPE: Double
  DEFAULT: 0.02
  OPTIONS: $n$ User-defined value.

• **REG_PARAM**
  Regularization parameter.
  TYPE: Double
  DEFAULT: 0.4
  OPTIONS: $n$ User-defined value.

• **SWAP_MO_INITIAL**
  This option must be used along with the SWAP_MO_FINAL option. SWAP_MO_INITIAL defines a vector for initial MO ordering. The vector should include the range of MOs that will be re-ordered.
  TYPE: Integer Array
  DEFAULT: [0]
  OPTIONS: User-defined.

• **SWAP_MO_FINAL**
  This option must be used along with the SWAP_MO_INITIAL option. SWAP_MO_FINAL defines a vector for final MO ordering.
  TYPE: Integer Array
  DEFAULT: [0]
  OPTIONS: User-defined.

• **CUTOFF**
  The cutoff value for numerical procedures.
  TYPE: Integer
  DEFAULT: 8
  OPTIONS: $n$ User-defined value.
• **EXTERNAL_NFRZV**  
  Externally defined frozen virtuals.  
  TYPE: Integer  
  DEFAULT: 0  
  OPTIONS: *n* User-defined value.

• **FOLLOW_CI_ROOT**  
  The electronic root that will be followed in an FCI computation.  
  TYPE: Integer  
  DEFAULT: 1  
  OPTIONS: *n* User-defined value.

• **FOLLOW_ROOT**  
  The electronic root that will be followed in a QDPT computation.  
  TYPE: Integer  
  DEFAULT: 1  
  OPTIONS: *n* User-defined value.

• **NACTMO**  
  Number of active MOs in the model space.  
  TYPE: Integer  
  DEFAULT: 2  
  OPTIONS: *n* User-defined value.

• **NALPHA**  
  Number of active alpha electrons in the model space.  
  TYPE: Integer  
  DEFAULT: 1  
  OPTIONS: *n* User-defined value.

• **NBETA**  
  Number of active beta electrons in the model space.  
  TYPE: Integer  
  DEFAULT: 1  
  OPTIONS: *n* User-defined value.

• **NSWAPMO**  
  Number of MOs to be swapped in the initial guess.  
  TYPE: Integer  
  DEFAULT: 0  
  OPTIONS: *n* User-defined value.

• **PRINT_LEVEL**  
  The amount of information to print to the output file.  
  TYPE: Integer  
  DEFAULT: 0  
  OPTIONS: *n* User-defined value.
- **QDPT_MAXITER**
  Maximum number of iterations.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: $n$ User-defined value.

- **CIVECS_READ**
  Do read CI coefficients from external files?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **CIVECS_WRITE**
  Do write CI coefficients to external files?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **CI_VECS**
  Do print CI coefficients?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **COMPUTE_S2**
  Do compute spin expectation value?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **CS_SINGLET**
  Do use closed-shell singlet wavefunction?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **FOCK_TYPE**
  Type of the Fock.
  TYPE: String
  DEFAULT: CORE
  OPTIONS: CORE, UHF, MCSCF, CASCI

- **FOLLOW_STATE**
  Follow what state, default is the ground state.
  TYPE: String
  DEFAULT: GROUND
  OPTIONS: GROUND, SINGLET, TRIPLET
• **MODEL_SPACE**
  How to select model space. It can be automatically determined via UNO or it can be
defined by users.
  TYPE: String
  DEFAULT: USER
  OPTIONS: USER, UNO

• **MO_READ**
  Do read MO coefficients from external files?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **MO_WRITE**
  Do write MO coefficients to external files?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **NAT_ORBS**
  Do use MP2 NOs?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **OS_SINGLET**
  Do use open-shell singlet wavefunction?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **PRINT_CONFIGS**
  Do print configurations?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **REGULARIZATION**
  Do use regularized denominators?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **SC_MO**
  Do semi-canonicalize ROHF MOs?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE
• **SYMMETRIC HEFF**
  Do symmetrize effective Hamiltonian?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **UNO**
  Do form UHF natural orbitals?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

• **REF WFN TYPE**
  Type of the ref wavefunction.
  TYPE: String
  DEFAULT: SCF
  OPTIONS: SCF

13 EOMEE : Equation-of-Motion Coupled-Cluster

**Code Authors:** A. Ünal and U. Bozkaya. MacroQC includes EOM-CCD, EOM-OCCD, and EOM-CCSD implementations with the density-fitting approach [84, 85].

13.1 Input Examples

*Single point eom-ccsd/cc-pvdz computation for 5 roots:*

```
$set_molecule H2O
  0 1
  O  0.00000  0.00000  -0.06577
  H  0.00000  -0.75906  0.52195
  H  0.00000   0.75906  0.52195
$end_molecule

$setGlobals
  basis cc-pvdz
  aux_basisscf cc-pvdz-jkfit
  aux_basiscorr cc-pvdz-ri
  method eom-ccsd
  reference rhf
  jobtype energy
  freeze_core true
$end_globals
```
13.2 EOMEE Module Options

- **NUM_ROOTS**
  Number of excited states for EOM computations.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: \( n \) User-defined value.

- **EXCITATION_RANGE**
  The gap between the occupied and virtual spaces from which initially excited state guesses are directed to the Davidson algorithm. The default value of 2 corresponds to HOMO-1, HOMO, LUMO, and LUMO+1.
  TYPE: Integer
  DEFAULT: 2
  OPTIONS: \( n \) User-defined value.

- **EOM_MAXITER**
  The maximum number of EOM iterations.
  TYPE: Integer
  DEFAULT: 80
  OPTIONS: \( n \) User-defined value.

- **MAX_VECS**
  Maximum dimension of the subspace.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: \( n \) User-defined value.

- **VECS_PER_ROOT**
  Subspace vectors used per root.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: \( n \) User-defined value.

- **SCHMIDT_ADD_RES_TOL**
  Tolerance value to add a guess vector of a root to the Davidson algorithm.
  TYPE: Double
  DEFAULT: 1.0e-3
  OPTIONS: \( n \) User-defined tolerance value.

- **EOM_E_CONVERGENCE**
  Convergence criterion for excitation energy.
• **EOM_RES_PRECONDITION_TOL**
  Tolerance value of EOM residual preconditioning.
  TYPE: Double
  DEFAULT: 1.0e-4
  OPTIONS: $n$ User-defined tolerance value.

• **EOMCC_GUESS**
  Specifies the initial guess of EOM-CC computations.
  TYPE: String
  DEFAULT: CIS
  OPTIONS: CIS, UNIT
  – CIS: The initial guess is generated from the configuration interaction singles method.
  – UNIT: The initial guess is generated from unit vectors.

• **UNIT_GUESS_TYPE**
  Type of initial guesses.
  TYPE: String
  DEFAULT: GEN
  OPTIONS: LIMITED, GEN
  – LIMITED: This option requests a limited guess to be generated around HOMO-LUMO orbitals. The dimension of guess is determined by the EXCITATION_RANGE option.
  – GEN: All singly excited Slater determinants are used in generating an initial guess.

• **EOM_UNIT_GUESS_COEFF**
  Initial guess for unrestricted EOM computations.
  TYPE: String
  DEFAULT: MIX
  OPTIONS: MIX, SINGLET, TRIPLET
  – MIX: Use a mixture of singlet and triplet states.
  – SINGLET: Use singlet coefficients, which means $c_I^A = 1/\sqrt{2}$ and $c_i^a = 1/\sqrt{2}$.
  – TRIPLET: Use triplet coefficients, which means $c_I^A = 1/\sqrt{2}$ and $c_i^a = -1/\sqrt{2}$. 
14 Fragment: Molecular Fragmentation Approaches

Code Authors: U. Bozkaya and B. Ermiş.

14.1 Systematic Molecular Fragmentation (SMF)

The MacroQC software provides the LSSMF approach for all theoretical methods available in it. Further, our LSSMF code can also collaborate with several quantum chemistry software. The current version can be used with the Psi4 \[41\] and Q-Chem \[103\] packages only.

14.2 Input Examples

14.2.1 Serial Mode

This is an example of a single-point energy computation at the LSSMF-rhf/cc-pvdz level. In this example, level 3 is used for generating bonded fragments.

```
# TEST_LABEL: 3-3-dimethylheptane_level3
$set_molecule
0 1
C  -2.701034000  -1.655296000  0.075154000
C  -2.373842000  -0.295418000  -0.551340000
C  -1.121132000  0.456476000  -0.019192000
C  -1.062210000  1.813812000  -0.747749000
C  -1.257793000  0.707881000  1.495575000
C   0.149643000  -0.378389000  -0.331976000
C   1.503136000  0.209803000  0.088913000
C   2.682769000  -0.702310000  -0.270142000
C   4.039004000  -0.133276000  0.152781000
H  -3.611969000  -2.075889000  -0.381179000
H  -1.895007000  -2.391312000  -0.074579000
H  -2.886260000  -1.579045000  1.158788000
H  -2.256461000  -0.424469000  -0.420386000
H  -0.218422000  2.431448000  -0.402749000
H  -0.957409000  1.677367000  -1.837538000
H  -1.985366000  2.391184000  -0.572273000
H  -0.459662000  1.368810000  1.868629000
H  -2.220489000  1.194614000  1.726545000
H  -1.208561000  -0.228561000  2.074156000
H   0.047882000  -1.369086000  0.145378000
H   0.172883000  -0.570617000  -1.421514000
H   1.659232000  1.194490000  -0.386834000
H   1.514669000  0.391461000  1.178685000
```
In this example, the mol_frag true option requests a single-point LSSMF-SCF energy computation. The level 3 option in the lssmf block indicates that a level 3 fragmentation will be used for bonded fragments. The nb_cutoff_max 10.0 option indicates a value of 10.0 Å will be used for $\Delta_{nb}$. The mode serial option indicates that this job will be run in the serial mode, and a single output file will be formed. When we execute the above example, we will see the following options in the output file:

Natom: 29
Ngroup: 9
Nbfrag: 26
Nnonbfrag: 10
Charge & Multiplicity: 0 1
Nelectron: 74
LSSMF level: 3
Nonbonded level: 1
Print level: 0
R12 tolerance: 10.00
Delta single bond: 0.40
Delta double bond: 0.07
Delta triple bond: 0.25
Group files will be split into 0 parts
Bonded files will be split into 0 parts
Nonbonded files will be splitted into 0 parts
Nuclear Repulsion Energy: 487.8480383549

These are the basic information about the LSSMF approach, which means for the given molecule there are 9 groups, 26 bonded fragments, and 10 no-bonded fragments. This section also prints out the basic parameters that used in the LSSMF approach: level 3 is used for bonded fragments, level 1 is used for NB fragments, $\Delta_{nb}$ tolerance of 10.0 Å is used, etc.

When energies of all bonded fragments are completed, the following table is printed out. In this table, the label and the coefficient of each bonded group, as well as corresponding energies, are provided. At the end of the table, the total bonded energy is printed.

<table>
<thead>
<tr>
<th>Coeff</th>
<th>Energy</th>
<th>Coeff*Energy</th>
<th>Bonded Group Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-157.3064946038</td>
<td>-157.3064946038</td>
<td>G_1_2_3_4</td>
</tr>
<tr>
<td>1</td>
<td>-157.3055496928</td>
<td>-157.3055496928</td>
<td>G_1_2_3_5</td>
</tr>
<tr>
<td>1</td>
<td>-157.303834931</td>
<td>-157.303834931</td>
<td>G_1_2_3_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.3061863028</td>
<td>-157.3061863028</td>
<td>G_2_3_4_5</td>
</tr>
<tr>
<td>1</td>
<td>-157.3049150998</td>
<td>-157.3049150998</td>
<td>G_2_3_4_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.3052965489</td>
<td>-157.3052965489</td>
<td>G_2_3_5_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.3050898279</td>
<td>-157.3050898279</td>
<td>G_2_3_6_7</td>
</tr>
<tr>
<td>1</td>
<td>-157.3069522452</td>
<td>-157.3069522452</td>
<td>G_3_4_5_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.3051381643</td>
<td>-157.3051381643</td>
<td>G_3_4_6_7</td>
</tr>
<tr>
<td>1</td>
<td>-157.3057145289</td>
<td>-157.3057145289</td>
<td>G_3_5_6_7</td>
</tr>
<tr>
<td>1</td>
<td>-157.3074706011</td>
<td>-157.3074706011</td>
<td>G_3_6_7_8</td>
</tr>
<tr>
<td>1</td>
<td>-157.3078019201</td>
<td>-157.3078019201</td>
<td>G_6_7_8_9</td>
</tr>
<tr>
<td>-2</td>
<td>-118.2701325572</td>
<td>236.5402651145</td>
<td>G_1_2_3</td>
</tr>
<tr>
<td>-2</td>
<td>-118.2684223683</td>
<td>236.5368447365</td>
<td>G_2_3_4</td>
</tr>
<tr>
<td>-2</td>
<td>-118.2687952758</td>
<td>236.5375905517</td>
<td>G_2_3_5</td>
</tr>
<tr>
<td>-3</td>
<td>-118.2675117406</td>
<td>354.8025352217</td>
<td>G_2_3_6</td>
</tr>
<tr>
<td>-1</td>
<td>-118.2706308152</td>
<td>118.2706308152</td>
<td>G_3_4_5</td>
</tr>
<tr>
<td>-2</td>
<td>-118.2691546284</td>
<td>236.5383092567</td>
<td>G_3_4_6</td>
</tr>
<tr>
<td>-2</td>
<td>-118.2695519038</td>
<td>236.5391038076</td>
<td>G_3_5_6</td>
</tr>
<tr>
<td>-3</td>
<td>-118.2703538608</td>
<td>354.8110615825</td>
<td>G_3_6_7</td>
</tr>
<tr>
<td>-1</td>
<td>-118.2698076315</td>
<td>118.2698076315</td>
<td>G_6_7_8</td>
</tr>
<tr>
<td>3</td>
<td>-79.2318269995</td>
<td>-237.6954809985</td>
<td>G_2_3</td>
</tr>
<tr>
<td>1</td>
<td>-79.2336613594</td>
<td>-79.2336613594</td>
<td>G_3_4</td>
</tr>
<tr>
<td>1</td>
<td>-79.2340541054</td>
<td>-79.2340541054</td>
<td>G_3_5</td>
</tr>
<tr>
<td>3</td>
<td>-79.2325730649</td>
<td>-237.6977191947</td>
<td>G_3_6</td>
</tr>
<tr>
<td>-1</td>
<td>-40.1984657167</td>
<td>40.1984657167</td>
<td>G_3</td>
</tr>
</tbody>
</table>
Total bonded energy for SCF = -352.4867942522

Similarly, all nonbonded dimers, their monomers, and individual energies are printed out as below.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(hartree)</td>
<td>(hartree)</td>
<td>(hartree)</td>
<td>(hartree)</td>
<td>(kcal/mol)</td>
<td></td>
</tr>
<tr>
<td>-80.39565616</td>
<td>-40.19782045</td>
<td>-40.19788569</td>
<td>0.00004998</td>
<td>0.03136212</td>
<td>G_1_G_7</td>
</tr>
<tr>
<td>-80.39556601</td>
<td>-40.19782045</td>
<td>-40.19773805</td>
<td>-0.00000752</td>
<td>-0.04717279</td>
<td>G_1_G_8</td>
</tr>
<tr>
<td>-80.39573075</td>
<td>-40.19782045</td>
<td>-40.19790816</td>
<td>-0.00000214</td>
<td>-0.00134471</td>
<td>G_1_G_9</td>
</tr>
<tr>
<td>-80.39444135</td>
<td>-40.1972185</td>
<td>-40.19773805</td>
<td>0.00001855</td>
<td>0.01163874</td>
<td>G_2_G_8</td>
</tr>
<tr>
<td>-80.39463707</td>
<td>-40.19672185</td>
<td>-40.19790816</td>
<td>-0.00000706</td>
<td>-0.00443044</td>
<td>G_2_G_9</td>
</tr>
<tr>
<td>-80.39635460</td>
<td>-40.19846572</td>
<td>-40.19790816</td>
<td>0.00001928</td>
<td>0.01209736</td>
<td>G_3_G_9</td>
</tr>
<tr>
<td>-80.3951052</td>
<td>-40.19802493</td>
<td>-40.19773805</td>
<td>0.00005245</td>
<td>0.03291412</td>
<td>G_4_G_8</td>
</tr>
<tr>
<td>-80.3959110</td>
<td>-40.19802493</td>
<td>-40.19790816</td>
<td>-0.00000801</td>
<td>-0.00502865</td>
<td>G_4_G_9</td>
</tr>
<tr>
<td>-80.3956849</td>
<td>-40.19800026</td>
<td>-40.19773805</td>
<td>0.00004982</td>
<td>0.03126424</td>
<td>G_5_G_8</td>
</tr>
<tr>
<td>-80.39591563</td>
<td>-40.19800026</td>
<td>-40.19790816</td>
<td>-0.00000721</td>
<td>-0.00452506</td>
<td>G_5_G_9</td>
</tr>
</tbody>
</table>

Nonbonded interaction energies larger than 1.0 kcal/mol.

| Sum of nonbonded interaction energies for SCF => 0.000158133758 (hartree) 0.1 (kcal/mol) |
-----------------------------------------------------------------------------------------------
Finally, the total LSSMF energy is printed out separately as follows:

| Total energy => -352.486636118422 |

14.2.2 Mode Sow-1

This is an example of a single-point energy computation at the LSSMF-ccsd(t)/cc-pvdz level. In this example, level 3 is used for generating bonded fragments. For this example, bonded.inp, nonbonded.inp, and groups.inp files are written. Some important information required for the REAP mode will be written to the lssmf.chk file. Make sure the lssmf.chk file is not deleted.

# TEST_LABEL: n-heptane
$set_molecule
0 1
C   3.2171710000 0.8842760000 0.2852810000
H   2.5889210000 1.7437160000 -0.0003400000
H   3.2354180000 0.8393510000 1.3882070000
H   4.2428730000 1.1008270000 -0.0552130000
C   2.7017540000 -0.4287940000 1.3882070000
H   2.6997800000 -0.3550810000 1.3882070000
H   3.4122590000 -1.2373920000 -0.0552130000
C   1.3020380000 -1.2373920000 0.1668920000
H   1.0873170000 -1.8624800000 -0.2043850000
H   1.3039970000 -0.9190170000 0.2738120000
C   0.1666320000 0.0884510000 -0.2738120000
The lssmf.chk file is printed out as below.

<table>
<thead>
<tr>
<th>Dimer</th>
<th>Mono1</th>
<th>Mono2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>7</td>
</tr>
</tbody>
</table>

Coefficients ---> Label of bonded frags
When computations are completed for fragments generated with the mode sow option. Then, the same input file should be run with the mode reap option, which initiates the reading procedure for each output file and computes the LSSMF energies. When all fragment output files are processed, the following table is printed out. The same table will be prepared for all the methods you specified in METHOD_NAME array.

<table>
<thead>
<tr>
<th>Coeff</th>
<th>Energy</th>
<th>Bonded Group Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-157.97849436</td>
<td>G_1_2_3_4</td>
</tr>
<tr>
<td>1</td>
<td>-157.97823483</td>
<td>G_2_3_4_5</td>
</tr>
<tr>
<td>1</td>
<td>-157.978063574</td>
<td>G_3_4_5_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.978063574</td>
<td>G_4_5_6_7</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7796321247</td>
<td>G_2_3_4</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7800829289</td>
<td>G_3_4_5</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7800993839</td>
<td>G_4_5_6</td>
</tr>
</tbody>
</table>

Total bonded energy for CCSD(T) = -275.5752956135

---------------|---------------|---------------|--------------|--------------|-----------|
-80.77367284  | -40.38692204  | -40.38692204  | -0.00034577  | -0.21697453  | G_1_G_5   |
-80.77342760  | -40.38692204  | -40.38692204  | -0.00007761  | -0.04870338  | G_1_G_6   |
-80.77387251  | -40.38692204  | -40.38692204  | -0.00018392  | -0.11540783  | G_1_G_7   |
-80.77296019  | -40.38692204  | -40.38692204  | -0.00022444  | -0.14078333  | G_2_G_6   |
-80.77331843  | -40.38692204  | -40.38692204  | -0.00042072  | -0.02639750  | G_2_G_7   |
-80.77334675  | -40.38692204  | -40.38692204  | -0.00018468  | -0.11588552  | G_3_G_7   |

Nonbonded interaction energies larger than 1.0 kcal/mol.

Sum of nonbonded interaction energies for CCSD(T) => -0.000856481000 (hartree)
-0.5 (kcal/mol)

Total energy for CCSD(T) = -275.576152094486

14.2.3 Mode Sow-2

This is an example of a single-point energy computation at the LSSMF-ccsd(t)/cc-pvdz level. In this example, the PSI4 software is employed instead of MacroQC. In this example...
level 3 is used for generating bonded fragments. When an external software is used, the options for the external software should be provided in the **FRAG_EXTERNAL** block. **MacroQC** generates all necessary input files for the allowed external software.

```plaintext
$set_molecule
0 1
C 3.2171710000 0.8842760000 0.2852810000
H 2.5889210000 1.7437160000 -0.0003400000
H 3.2354180000 0.8393510000 1.3882070000
H 4.2428730000 1.1008270000 -0.0552130000
C 2.7017540000 -0.4287940000 -0.3125600000
H 2.6997800000 -0.3550810000 -1.4162340000
H 3.4122590000 -1.2373920000 -0.0652100000
C 1.3020380000 -0.8436370000 0.1668920000
H 1.0873170000 -1.8624800000 -0.2043850000
H 1.3039970000 -0.9190170000 1.2713390000
C 0.1666320000 0.0884510000 -0.2738120000
H 0.3379080000 1.1060480000 0.1216760000
H 0.1838900000 0.1848320000 -1.3764850000
C -1.2203000000 -0.3892860000 0.1693790000
H -1.2368210000 -0.4905490000 1.2714170000
H -1.4015240000 -1.4059910000 -0.2286320000
C -2.3613920000 0.5347540000 -0.2691010000
H -2.1816020000 1.5498180000 0.1311300000
H -2.3432560000 0.6371130000 -1.3700840000
C -3.7422770000 0.0473140000 0.1757680000
H -3.8046130000 -0.0302140000 1.2748300000
H -4.5400490000 0.7316460000 -0.1555780000
H -3.9662530000 -0.9510910000 -0.2374760000
$end_molecule

$setGlobals
  jobtype energy
  mol_frag true
$endGlobals

$setLssmf
  level 3
  nb_level 1
  method_name [scf;mp2;ccd;ccsd;ccsd(t)]
  mode sow
  software psi4
$endLssmf

$setFragExternal
```
According to the methods you choose in the METHOD_NAME option, your PSI4 input file will be prepared as follows:

# Psi4 Input File
memory 5 gb
SET {
  BASIS CC-PVDZ
  DF_BASIS_SCF CC-PVDZ-JKFIT
  DF_BASIS_CC CC-PVDZ-RI
  GUESS SAD
  REFERENCE RHF
  SCF_TYPE DF
  FREEZE_CORE TRUE
  QC_MODULE OCC
  MP2_TYPE DF
  MP_TYPE DF
  CC_TYPE DF
}
molecule G_1_2_3_4 {
  0 1
  C 3.2171710000 0.8842760000 0.2852810000 #G1
  H 2.5889210000 1.7437160000 -0.0003400000 #G1
  H 3.2354180000 0.8393510000 1.3882070000 #G1
  H 4.2428730000 1.1008270000 -0.0552130000 #G1
  C 2.7017540000 -0.4287940000 -0.3125600000 #G2
  H 2.6997800000 -0.3550810000 -1.4162340000 #G2
  H 3.4122590000 -1.2373920000 -0.0652510000 #G2
  C 1.3020380000 -0.8436370000 0.1668920000 #G3
  H 1.0873170000 -1.8624800000 -0.2043850000 #G3
  H 1.3039970000 -0.9190170000 1.2713390000 #G3
  C 0.1666320000 0.0884510000 -0.2738120000 #G4
  H 0.3379080000 1.1060480000 0.1216760000 #G4
  H 0.1838900000 0.1848320000 -1.3764850000 #G4
  H -0.8096951316 -0.2478507039 0.0381711382 #G4-CAP
noreorient
nocom
symmetry c1
14.2.4 Mode Sow-3

This is an example of a single-point energy computation at the LSSMF-mp2/cc-pvdz level. In this example, level 3 is selected with the nb_level 3 option. For this example, bonded.inp, dimers.inp, and monomers.inp files are written. Some important information required for the REAP mode will be written to the lssmf.chk file. Make sure the lssmf.chk file is not deleted.

# TEST_LABEL: n-heptane
$set_molecule
0 1
C 3.21717100000 0.88427600000 0.28528100000
H 2.58892100000 1.74371600000 -0.00034000000
H 3.23541800000 0.83935100000 1.38820700000
H 4.24287300000 1.10082700000 -0.05521300000
C 2.70175400000 -0.42879400000 -0.31256000000
H 2.69978000000 -0.35508100000 -1.41623400000
H 3.41225900000 -1.23739200000 -0.06525100000
C 1.30203800000 -0.84363700000 0.16689200000
H 1.08731700000 -1.86248000000 -0.20438500000
H 1.30399700000 -0.91901700000 1.27133900000
C 0.16663200000 0.08845100000 0.12167600000
H 0.33790800000 1.10604800000 0.12167600000
H 0.18389000000 0.18483200000 -1.37648500000
C -1.22030000000 -0.38928600000 0.16937900000
H -1.23682100000 -0.49054900000 1.27141700000
H -1.40152400000 -1.40599100000 -0.22863200000
C -2.36139200000 -0.53475400000 -0.26910100000
H -2.18160200000 1.54981800000 0.13113000000
H -2.34325600000 0.63711300000 -1.37008400000
C -3.74227700000 0.04731400000 0.17576800000
H -3.80461300000 -0.03021400000 1.27483000000
H -4.54004900000 0.73164600000 -0.15557800000
The lssmf.chk file is printed out as below.

<table>
<thead>
<tr>
<th>Dimer</th>
<th>Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &lt;-&gt; 5_6_7</td>
<td>1</td>
</tr>
<tr>
<td>7 &lt;-&gt; 1_2_3</td>
<td>1</td>
</tr>
<tr>
<td>1_2 &lt;-&gt; 6_7</td>
<td>1</td>
</tr>
<tr>
<td>1 &lt;-&gt; 6_7</td>
<td>-1</td>
</tr>
<tr>
<td>7 &lt;-&gt; 1_2</td>
<td>-1</td>
</tr>
</tbody>
</table>

Coefficients of bonded frags:
1 --> 1
2 --> 1
3 --> 1
4 --> 1
5 --> -1
6 --> -1
7 --> -1

Coefficients ---> Label of bonded frags
********** ********************
When computations are completed for fragments generated with the mode sow option. Then, the same input file should be run with the mode reap option, which initiates the reading procedure for each output file and computes the LSSMF energies. When all fragment output files are processed, the following table is printed out. The same table will be prepared for all the methods you specified in the METHOD_NAME array.

---

<table>
<thead>
<tr>
<th>Coeff</th>
<th>Energy</th>
<th>Coeff*Energy</th>
<th>Bonded Group Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-157.8967782614</td>
<td>-157.8967782614</td>
<td>G_1_2_3_4</td>
</tr>
<tr>
<td>1</td>
<td>-157.8965730873</td>
<td>-157.8965730873</td>
<td>G_2_3_4_5</td>
</tr>
<tr>
<td>1</td>
<td>-157.8970383126</td>
<td>-157.8970383126</td>
<td>G_3_4_5_6</td>
</tr>
<tr>
<td>1</td>
<td>-157.8978930447</td>
<td>-157.8978930447</td>
<td>G_4_5_6_7</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7158761433</td>
<td>118.7158761433</td>
<td>G_2_3_4</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7163633621</td>
<td>118.7163633621</td>
<td>G_3_4_5</td>
</tr>
<tr>
<td>-1</td>
<td>-118.7164024508</td>
<td>118.7164024508</td>
<td>G_4_5_6</td>
</tr>
</tbody>
</table>

Total bonded energy for MP2 = -275.4396407497

---

<table>
<thead>
<tr>
<th>Energy of Dimer</th>
<th>Energy of Mono1</th>
<th>Energy of Mono2</th>
<th>Nonbonded I.E.</th>
<th>Coeff</th>
<th>I.E. * Coeff</th>
<th>Dimer Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>(hartree)</td>
<td>(hartree)</td>
<td>(hartree)</td>
<td>(hartree)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-159.07701523</td>
<td>-40.35941955</td>
<td>-118.71718008</td>
<td>-0.00041560</td>
<td>1</td>
<td>-0.26</td>
<td>G_1_G_5_6_7</td>
</tr>
<tr>
<td>-159.07620847</td>
<td>-40.35942736</td>
<td>-118.71658498</td>
<td>-0.00022613</td>
<td>1</td>
<td>-0.14</td>
<td>G_1_G_5_6_7</td>
</tr>
<tr>
<td>-159.07335335</td>
<td>-79.53647925</td>
<td>-79.53659144</td>
<td>-0.00028266</td>
<td>1</td>
<td>-0.18</td>
<td>G_7_G_1_2_3</td>
</tr>
<tr>
<td>-119.89610130</td>
<td>-40.35941955</td>
<td>-118.71658498</td>
<td>-0.00009301</td>
<td>-1</td>
<td>0.06</td>
<td>G_1_G_6_7</td>
</tr>
<tr>
<td>-119.89598802</td>
<td>-40.35942736</td>
<td>-79.53647925</td>
<td>-0.00008141</td>
<td>-1</td>
<td>0.03</td>
<td>G_7_G_1_2</td>
</tr>
</tbody>
</table>

Sum of nonbonded interaction energies for MP2 = -0.0007826705 (hartree)
-0.5 (kcal/mol)

---

Total energy for MP2 = -275.440423420194

---

### 14.2.5 Assigning Charge to Groups

This is an example of a single-point energy computation at the LSSMF-mp2/cc-pvdz level. In this example, level 3 is selected with the nb_level 1 option. CARBAMINO_CHARGE INPUT_FCHARGE option must be entered to specify the charge for each group in the input file.

# TEST_LABEL: C5H10N2O4
$set_molecule
$end_molecule

$set_globals
    jobtype energy
    basis cc-pvdz
    aux_basis_corr cc-pvdz-ri
    aux_basis_scf cc-pvdz-jkfit
    method mp2
    mol_frag true
    memory 5000
$end_globals

$set_lssmf
    level 3
    nb_level 1
    method_name [scf;mp2]
    mode sow
    delta_sb 0.40
    delta_db 0.07
    delta_tb 0.25
    carbamino_charge input_fcharge
$end_lssmf
The `groups.inp` file will appear as follows.

```
$SET_GLOBALS
MOL_FRAG FALSE
MEMORY 5000
AUX_BASIS_CORR CC-PVDZ-RI
AUX_BASIS_SCF CC-PVDZ-JKFIT
BASIS CC-PVDZ
JOBTYPE ENERGY
METHOD MP2
$END_GLOBALS

$SET_LSSMF
DELTA_DB 0.07
DELTA_SB 0.4
DELTA_TB 0.25
LEVEL 3
NB_LEVEL 1
CARBAMINO_CHARGE INPUT_FCHARGE
MODE SOW
METHOD_NAME [SCF;MP2]
$END_LSSMF

$SET_MOLECULE G1
  -1 1
  O  -9.0681100000 -0.7226600000 -1.5197800000
  O  -7.0119000000 -0.0295700000 -1.1953800000
  C  -8.3203400000 -0.2399000000 -1.4390500000
  H  -8.6755307237 -1.2415959868 -1.5821484211 # CAP
$END_MOLECULE

@@@

$SET_MOLECULE G2
  1 1
  N  -8.4720600000  -2.4901800000 -0.4778900000
  H  -9.0792300000 -2.2930200000  0.3470100000
  H  -7.4773400000 -2.3207700000 -0.2028500000
  H  -8.5059300000 -3.4986900000 -0.7304100000
  H  -8.7168946939 -1.9161281633 -1.2858687755 # CAP
$END_MOLECULE

@@@

$SET_MOLECULE G3
  0 1
  C  -8.8249100000  -1.6628700000 -1.6423300000
  H  -8.2519900000 -2.0505100000 -2.5152400000
  H  -8.4697192763 -0.6611740132 -1.4992315789 # CAP
  H  -8.5680736054 -2.2650616327 -0.7947444218 # CAP
  H  -9.8986761184 -1.6942942105 -1.9193262500 # CAP
$END_MOLECULE
```
### 14.3 FRAGMENT Module Options

- **MODE**
  Type of the mode.
  TYPE: String
  DEFAULT: SOW
OPTIONS: SOW, REAP, SERIAL, SPLIT

- SOW: Forms fragment files for mode sow computations. For nonbonded level 1, bonded.inp, nonbonded.inp, and groups.inp files are written. For nonbonded level 3, bonded.inp, dimers.inp, and monomers.inp files are written. In case of a large number of fragments, users may ask further split each file into the desired number of input files. For this purpose, the npiece_b, npiece_nb, and npiece_g options can be used.

- REAP: When computations are completed for fragments generated with the SOW option, REAP reads output files to compute the LSSMF energies. Note that when MacroQC program is executed with the REAP option, all output files should be available in the working directory.

- SERIAL: It allows the LSSMF computations to run in serial mode. In this case, all fragments are generated on the fly, and a single output file is formed.

- SPLIT: Splits fragment input files created with the SOW option into the given number of files. This option should be used with the npiece_b, npiece_nb, and npiece_g options.

**NBONDED_CUTOFF_TYPE**
Type of nonbonded cutoff.
TYPE: String
DEFAULT: DCRR
OPTIONS: DCRR, DBE

  - DCRR: The ratio of distance to covalent radii
  - DBE: Distance-based elimination.

**SOFTWARE**
For which quantum chemistry software input files will be prepared? This option must be used along with the MODE SOW and MODE REAP options.
TYPE: String
DEFAULT: MACROQC
OPTIONS: MACROQC, PSI4, QCHEM

**METHOD_NAME**
Which methods do you want to integrate the LSSMF approach? It works with the REAP and SOW options. The same string should be used for the REAP and SOW options. This option is coded for MacroQC and PSI4 software. This option accepts multiple method names since energies of several methods can be available in output files. Use "," character to specify multiple method names, such as [SCF;MP2;CCD]. For QCHEM software, see the ENERGY_LINE option.
TYPE: String Array
DEFAULT: [SCF]
OPTIONS: User defined method(s) name.
• **ENERGY_LINE**
  The string before energy values can be specified here so that MacroQC grep the energy from outputs of external software. It works with the REAP option. If you are working with the ENERGY_LINE option, you do not need this option.
  TYPE: LINE
  DEFAULT: "Total Energy"
  OPTIONS: User defined energy line.

• **CARBAMINO_CHARGE**
  Specification of charge of carboxyl and amino groups in protein structures.
  TYPE: STRING
  DEFAULT: AUTO
  OPTIONS: AUTO, INPUT_FCHARGE
    - AUTO: The charge of the carboxyl and amino groups are determined atomically.
    - INPUT_FCHARGE: The charge of the carboxyl and amino groups is imported from the input file.

• **PRINT_LEVEL**
  The amount of information to print to the output file.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.

• **NPIECE_B**
  Specifies the number of ASCII files that the bonded fragments file will be split. Used with SOW or SPLIT options. The same value should be used for the SOW and REAP options.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.

• **NPIECE_NB**
  Specifies the number of ASCII files that the nonbonded fragments file will be split. Used with SOW or SPLIT options. The same value should be used for the SOW and REAP options.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.

• **NPIECE_G**
  Specifies the number of ASCII files that the groups’ file will be split. Used with SOW or SPLIT options. The same value should be used for the SOW and REAP options.
  TYPE: Integer
  DEFAULT: 0
  OPTIONS: n User-defined value.
• **LEVEL**
  Fragmentation level for bonded fragments. For accurate results, at least level 3 should be chosen.
  TYPE: Integer
  DEFAULT: 3
  OPTIONS: 1, 2, 3, 4, 5

• **NB_LEVEL**
  Fragmentation level for nonbonded fragments.
  TYPE: Integer
  DEFAULT: 1
  OPTIONS: 1, 3

• **NCIE_PRINT_TOL**
  Threshold value (in kcal mol\(^{-1}\)) to print noncovalent interaction energies to the output file.
  TYPE: Integer
  DEFAULT: 1.0
  OPTIONS: User-defined value.

• **NB_CUTOFF_MAX**
  Tolerance value for the DCRR and DBE algorithms. If the distance between the closest atoms of two groups is larger than **NB_CUTOFF_MAX**, then this nonbonded fragment is disregarded.
  TYPE: Integer
  DEFAULT: 5.0
  OPTIONS: User-defined value.

• **DELTA_SB**
  Tolerance value (in Å) to identify an inter-atomic distance as a single bond.
  TYPE: Integer
  DEFAULT: 0.40
  OPTIONS: User-defined value.

• **DELTA_DB**
  Tolerance value (in Å) to identify an inter-atomic distance as a double bond.
  TYPE: Integer
  DEFAULT: 0.05
  OPTIONS: User-defined value.

• **DELTA_TB**
  Tolerance value (in Å) to identify an inter-atomic distance as a triple bond.
  TYPE: Integer
  DEFAULT: 0.25
  OPTIONS: User-defined value.
References


[37] H. B. Schlegel, WIREs Computational Molecular Science 1, 790 (2011).


