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

User's Manual

Uğur Bozkaya,[a] Betül Ermiş,[a] Yavuz Alagöz,[a] Aslı Ünal,[a] and Ali Kaan Uyar[a]

[a] *Department of Chemistry, Hacettepe University, Ankara 06800, Turkey*

MacroQC Version: 1.0.7
Created on: July 28, 2023
Git Version: bb1422ac
Report bugs to: macroqcprog@gmail.com

Contents

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 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19] 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 [20]. 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.

Method	**Energy**	**Gradient**[a]
SCF	RHF | UHF | ROHF	RHF | UHF
MP2	RHF | UHF | ROHF	RHF | UHF
MP2.5	RHF | UHF	RHF | UHF
MP3	RHF | UHF	RHF | UHF
CCD	RHF | UHF | ROHF	RHF | UHF
LCCD	RHF | UHF | ROHF	RHF | UHF
CCSD	RHF | UHF | ROHF	RHF | UHF
CCSD(T)	RHF | UHF | ROHF	RHF | UHF
CCSD(T)$_\Lambda$	RHF | UHF | ROHF	
OMP2	RHF | UHF | ROHF	RHF | UHF
OMP2.5	RHF | UHF | ROHF	RHF | UHF
OMP3	RHF | UHF | ROHF	RHF | UHF
OLCCD	RHF | UHF | ROHF	RHF | UHF
OCCD	RHF | UHF | ROHF	RHF | UHF
OCCD(T)	RHF | UHF | ROHF	
OCCD(T)$_\Lambda$	RHF | UHF | ROHF	
FNO-MP2.5	RHF | UHF	
FNO-MP3	RHF | UHF	
FNO-CCD	RHF | UHF | ROHF	
FNO-LCCD	RHF | UHF | ROHF	
FNO-CCSD	RHF | UHF | ROHF	
FNO-CCSD(T)	RHF | UHF | ROHF	
FNO-CCSD(T)$_\Lambda$	RHF | UHF | ROHF	
EOM-CCSD	RHF | UHF	
EOM-CCD	RHF | UHF	
EOM-OCCD	RHF | UHF	
QDPT2	UHF | ROHF	
CAS-CI	UHF | ROHF	
CIS	RHF | UHF	
FCI	UHF | ROHF	
LSSMF-SCF	RHF | UHF | ROHF	
LSSMF-MP2	RHF | UHF | ROHF	
LSSMF-MP2.5	RHF | UHF	
LSSMF-MP3	RHF | UHF	
LSSMF-CCD	RHF | UHF | ROHF	
LSSMF-LCCD	RHF | UHF | ROHF	
LSSMF-CCSD	RHF | UHF | ROHF	

LSSMF-CCSD(T)	RHF | UHF | ROHF	
LSSMF-CCSD(T)$_\Lambda$	RHF | UHF | ROHF	
LSSMF-OMP2	RHF | UHF | ROHF	
LSSMF-OMP2.5	RHF | UHF | ROHF	
LSSMF-OMP3	RHF | UHF | ROHF	
LSSMF-OLCCD	RHF | UHF | ROHF	
LSSMF-OCCD	RHF | UHF | ROHF	
LSSMF-OCCD(T)	RHF | UHF | ROHF	
LSSMF-OCCD(T)$_\Lambda$	RHF | UHF | ROHF	
LSSMF-FNO-MP2.5	RHF | UHF	
LSSMF-FNO-MP3	RHF | UHF	
LSSMF-FNO-CCD	RHF | UHF | ROHF	
LSSMF-FNO-LCCD	RHF | UHF | ROHF	
LSSMF-FNO-CCSD	RHF | UHF | ROHF	
LSSMF-FNO-CCSD(T)	RHF | UHF | ROHF	
LSSMF-FNO-CCSD(T)$_\Lambda$	RHF | UHF | ROHF	

[a] For all gradient methods, ionization potentials via Extended Koopmans' Theorem (EKT) are available.

2 Citing MacroQC

2.1 Overall MacroQC Package

- Bozkaya, U.; Ermiş, B.; Alagöz, Y.; Ünal, A.; Uyar, A. K. 2022. "MacroQC 1.0: An Electronic Structure Theory Software for Large-Scale Applications". *J. Chem. Phys.* 156, 044801. DOI: 10.1063/5.0077823

2.2 Orbital-Optimized Post-Hartree-Fock Methods

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

- Bozkaya, U. 2014. "Analytic Energy Gradients and Spin Multiplicities for Orbital-Optimized Second-Order Perturbation Theory with Density-Fitting Approximation: An Efficient Implementation". *J. Chem. Theory Comput.* 10, 4389-4399. DOI: 10.1021/ct500634s

- Bozkaya, U. 2014. "Orbital-Optimized Second-Order Perturbation Theory with Density-Fitting and Cholesky Decomposition Approximations: An Efficient Implementation". *J. Chem. Theory Comput.* 10, 2371-2378. DOI: 10.1021/ct500231c

- Bozkaya, U.; Sherrill, C. D. 2013. "Analytic Energy Gradients for the Orbital-Optimized Second-Order Møller-Plesset Perturbation Theory". *J. Chem. Phys.* 138, 184103. DOI: 10.1063/1.4803662

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

- Bozkaya, U. 2018. "Analytic Energy Gradients for Orbital-Optimized MP3 and MP2.5 with the Density-Fitting Approximation: An Efficient Implementation". *J. Comput. Chem.* 39, 351-360. DOI: 10.1002/jcc.25122

- Bozkaya, U. 2016. "Orbital-Optimized MP3 and MP2.5 with Density-Fitting and Cholesky Decomposition Approximations". *J. Chem. Theory Comput.* 12, 1179-1188. DOI: 10.1021/acs.jctc.5b01128

- Soydaş, E.; Bozkaya, U. 2015. "Assessment of Orbital-Optimized MP2.5 for Thermochemistry and Kinetics: Dramatic Failures of Standard Perturbation Theory Approaches for Aromatic Bond Dissociation Energies and Barrier Heights of Radical Reactions". *J. Chem. Theory Comput.* 11, 1564-1573. DOI: 10.1021/ct501184w

- Bozkaya, U.; Sherrill, C. D. 2014. "Orbital-optimized MP2.5 and Its Analytic Gradients: Approaching CCSD(T) Quality for Noncovalent Interactions". *J. Chem. Phys.* 141, 204105. DOI: 10.1063/1.4902226

- Bozkaya, U. 2013. "Analytic Energy Gradients for the Orbital-Optimized Third-Order Møller-Plesset Perturbation Theory". *J. Chem. Phys.* 139, 104116. DOI: 10.1063/1.4820877

- Soydaş, E.; Bozkaya, U. 2013. "Assessment of Orbital-Optimized Third-Order Møller-Plesset Perturbation Theory and Its Spin-Component and Spin-Opposite Scaled Variants for Thermochemistry and Kinetics". *J. Chem. Theory Comput.* 9, 1452-1460. DOI: 10.1021/ct301078q

- Bozkaya, U. 2011. "Orbital-Optimized Third-Order Møller-Plesset Perturbation Theory and Its Spin-Component and Spin-Opposite Scaled Variants: Application to Symmetry Breaking Problems". *J. Chem. Phys.* 135, 224103. DOI: 10.1063/1.3665134

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

- Bozkaya, U. 2016. "Orbital-Optimized Linearized Coupled-Cluster Doubles with Density-Fitting and Cholesky Decomposition Approximations: An Efficient Implementation". Phys. Chem. Chem. Phys. 18, 11362-11373. DOI: 10.1039/C6CP00164E

- Soydaş, E.; Bozkaya, U. 2014. "Assessment of the Orbital-Optimized Coupled-Electron Pair Theory for Thermochemistry and Kinetics: Improving upon CCSD and CEPA(1)". *J. Comput. Chem.* 35 , 1073-1081. DOI: 10.1002/jcc.23592

- Bozkaya, U.; Sherrill, C. D. 2013. "Orbital-Optimized Coupled-Electron Pair Theory and its Analytic Gradients: Accurate Equilibrium Geometries, Harmonic Vibrational Frequencies, and Hydrogen Transfer Reactions". *J. Chem. Phys.* 139, 054104. DOI: 10.1063/1.4816628

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

- Bozkaya, U.; Ünal, A.; Alagöz, Y. 2020. "Energy and Analytic Gradients for the Orbital-Optimized Coupled-Cluster Doubles Method with the Density-Fitting Approximation: An Efficient Implementation", *J. Chem. Phys.* 153, 244115. DOI: 10.1063/5.0035811

- Bozkaya, U.; Turney, J. M.; Yamaguchi, Y.; Schaefer, H. F.; Sherrill, C. D. 2011. "Quadratically Convergent Algorithm for Orbital Optimization in the Orbital-Optimized Coupled-Cluster Doubles Method and in Orbital-Optimized Second-Order Møller-Plesset Perturbation Theory". *J. Chem. Phys.* 135, 104103. DOI: 10.1063/1.3631129

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

- Alagöz, Y.; Ünal, A.; Bozkaya, U. 2021. "Efficient Implementations of the Symmetric and Asymmetric Triple Excitation Corrections for the Orbital-Optimized Coupled-Cluster Doubles Method with the Density-Fitting Approximation", *J. Chem. Phys.* (submitted).

2.3 Perturbation Theory (PT)

- Bozkaya, U. 2018. "Analytic energy gradients for orbital-optimized MP3 and MP2.5 with the density-fitting approximation: An efficient implementation". *J. Comput. Chem.* 39, 351-360. DOI: 10.1002/jcc.25122

- Bozkaya, U. 2016. "Orbital-Optimized MP3 and MP2.5 with Density-Fitting and Cholesky Decomposition Approximations". *J. Chem. Theory Comput.* 12, 1179-1188. DOI: 10.1021/acs.jctc.5b01128

- Bozkaya, U. 2014. "Derivation of General Analytic Gradient Expressions for Density-Fitted Post-Hartree-Fock Methods: An Efficient Implementation for the Density-Fitted Second-Order Møller-Plesset Perturbation Theory". *J. Chem. Phys.* 141, 124108. DOI: 10.1063/1.4903269

- Bozkaya, U.; Sherrill, C. D. 2014. "Orbital-optimized MP2.5 and Its Analytic Gradients: Approaching CCSD(T) Quality for Noncovalent Interactions". *J. Chem. Phys.* 141, 204105. DOI: 10.1063/1.4902226

2.4 Coupled-Cluster (CC)

- Bozkaya, U.; Sherrill, C. D. 2017. "Analytic energy gradients for the coupled-cluster singles and doubles with perturbative triples method with the density-fitting approximation". *J. Chem. Phys.* 147, 044104. DOI: 10.1063/1.4994918

- Bozkaya, U. 2016. "A Noniterative Asymmetric Triple Excitation Correction for The Density-Fitted Coupled-Cluster Singles and Doubles Method: Preliminary Applications". *J. Chem. Phys.* 144, 144108. DOI: 10.1063/1.4945706

- Bozkaya, U.; Sherrill, C. D. 2016. "Analytic Energy Gradients for the Coupled-Cluster Singles and Doubles Method with the Density-Fitting Approximation". *J. Chem. Phys.* 144, 174103. DOI: 10.1063/1.4948318

- Bozkaya, U. 2014. "Derivation of General Analytic Gradient Expressions for Density-Fitted Post-Hartree-Fock Methods: An Efficient Implementation for the Density-Fitted Second-Order Møller-Plesset Perturbation Theory". *J. Chem. Phys.* 141, 124108. DOI: 10.1063/1.4903269

- Soydaş, E.; Bozkaya, U. 2014. "Assessment of the Orbital-Optimized Coupled-Electron Pair Theory for Thermochemistry and Kinetics: Improving upon CCSD and CEPA(1)". *J. Comput. Chem.* 35, 1073-1081. DOI: 10.1002/jcc.23592

- Bozkaya, U.; Sherrill, C. D. 2013. "Orbital-Optimized Coupled-Electron Pair Theory and its Analytic Gradients: Accurate Equilibrium Geometries, Harmonic Vibrational Frequencies, and Hydrogen Transfer Reactions". *J. Chem. Phys.* 139, 054104. DOI: 10.1063/1.4816628

- Bozkaya, U.; Schaefer, H. F. 2012 "Symmetric and Asymmetric Triple-Excitation Corrections for Orbital-Optimized Coupled-Cluster Doubles Method: Improving Upon CCSD(T) and CCSD(T)Λ. Application to Potential Energy Surfaces". *J. Chem. Phys.* 136, 204114. DOI: 10.1063/1.4720382

2.5 Second-Order Quasidegenerate Perturbation Theory (QDPT2)

- Servan, S. A.; Ünal, A.; Hamarat, B.; Bozkaya, U. 2020. "Assessment of the Density-Fitted Second-Order Quasidegenerate Perturbation Theory for Transition Energies: Accurate Computations of Singlet-Triplet Gaps for Charge-Transfer Compounds", *J. Phys. Chem. A* 124, 6889-6898. DOI: 10.1021/acs.jpca.0c04555

- Bozkaya, U. 2019. "An Efficient Implementation of the Second-Order Quasidegenerate Perturbation Theory with Density-Fitting and Cholesky Decomposition Approximations: Is It Possible to Use HartreeFock Orbitals for a Multiconfigurational Perturbation Theory?". *J. Chem. Theory Comput.* 15, 4415-4429.
DOI: 10.1021/acs.jctc.9b00378

2.6 Extended Koopmans' Theorem (EKT)

- Bozkaya, U.; Ünal, A. 2018. "State-of-the-Art Computations of Vertical Ionization Potentials with the Extended Koopmans' Theorem Integrated with the CCSD(T) Method". *J. Phys. Chem. A* 122, 4375-4380. DOI: 10.1021/acs.jpca.8b01851

- Bozkaya, U. 2014. "Accurate Electron Affinities from the Extended Koopmans' Theorem Based on Orbital-Optimized Methods". *J. Chem. Theory Comput.* 10, 2041-2048. DOI: 10.1021/ct500186j

- Bozkaya, U. 2013. "The Extended Koopmans' Theorem for Orbital-Optimized Methods: Accurate Computation of Ionization Potentials". *J. Chem. Phys.* 139, 154105. DOI: 10.1063/1.4825041

2.7 Molint Framework

- Bozkaya, U. 2021. "Molint 1.0: A Framework for the Computation of Molecular Integrals and Their Derivatives for Density-fitted Methods", *Int. J. Quantum Chem.* 121, e26623. DOI: 10.1002/qua.26623

2.8 Dipole Moments and One-Electron Properties

- Bozkaya, U.; Soydaş, E.; Filiz, B. 2020. "State-of-the-Art Computations of Dipole Moments Using Analytic Gradients of High-Level Density-Fitted Coupled-Cluster Methods with Focal-Point Approximations". *J. Comput. Chem.* 41, 769-779. DOI: 10.1002/jcc.26126

2.9 Anharmonic Force Field and IR Spectra

- Ermiş, B.; Ünal, A.; Soydaş, E.; Bozkaya, U. 2021. "Anharmonic Force Field from Coupled-Cluster Methods and Accurate Computation of Infrared Spectra", *Adv. Quantum Chem.* 83, 139-153. DOI: 10.1016/bs.aiq.2021.05.003.

2.10 Linear-Scaling Systematic Molecular Fragmentation

- Bozkaya, U.; Ermiş, B. 2020. Linear-Scaling Systematic Molecular Fragmentation Approach for High-Level Coupled-Cluster Methods: Coupled-Cluster Meets Macromolecules. https://doi.org/10.26434/chemrxiv.12702425.v2.

2.11 Equation-of-Motion Coupled-Cluster

- Ünal A.; Bozkaya, T.; Bozkaya, U. 2021. An Efficient Implementation of Equation-of-Motion Coupled-Cluster an Enhanced Algorithm for the Particle-Particle Ladder Term Using a Hybrid Density-Fitting/Cholesky Decomposition Approach, (unpublished).

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.

```
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 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.

```
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 take 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 (macroqcprog@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:

```
export MQC_SCRATCH=/home/user1/custom_scrdir
export MQCDATADIR=/home/user1/custom_datadir
```

Examples for `csh` and `tcsh`:

```
setenv MQC_SCRATCH /home/user1/custom_scrdir
setenv MQCDATADIR /home/user1/custom_datadir
```

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.

Section Start	Section End	Description
$set_molecule	$end_molecule	Contains charge, multiplicity, and the molecular coordinate input (mandatory).
$set_globals	$end_globals	Global options.
$set_basis	$end_basis	Assign primary basis sets for individual atoms.
$set_aux_basis_scf	$end_aux_basis_scf	Assign JK-Fit type auxiliary basis sets for individual atoms.
$set_aux_basis_corr	$end_aux_basis_corr	Assign RI-type auxiliary basis sets for individual atoms.
$set_sad_aux_basis	$end_sad_aux_basis	Assign the auxiliary basis set for the SAD guess.
$set_opt	$end_opt	Options for the opt module.
$set_freq	$end_freq	Options for the freq module.
$set_scf	$end_scf	Options for the scf module.
$set_dfocc	$end_dfocc	Options for the dfocc module.
$set_qdpt	$end_qdpt	Options for the qdpt module.
$set_eomee	$end_eomee	Options for the eomee module.
$set_lssmf	$end_lssmf	Options for the fragment module.
$set_frag_external	$end_frag_external	Options for the external software used for fragment module.
$set_dg_external	$end_dg_external	Options for the external software used for freq module.

* 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:

```
# 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:

```
$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
```

```
$end_globals
```

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:

```
$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     1.04846012
 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   0.00003690    -0.00022249    -0.00011859 1.0
$end_molecule

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

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:

```
$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

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

```
    reference rhf
    jobtype energy
$end_globals

@@@

$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

$set_globals
    basis cc-pvtz
    aux_basis_scf cc-pvtz-jkfit
    aux_basis_corr cc-pvtz-ri
    method mp2
    reference rhf
    jobtype opt
$end_globals

@@@
```

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.

- **STEP_BOHR**
  Specifies the step-size for numerical differentiation (in atomic units). The same value should be used for the `SOW` and `REAP` options of `freq` module.
  TYPE: Double
  DEFAULT: 0.05
  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.
- EOM-CCD: Equation-of-motion coupled-cluster doubles.
- EOM-OCCD: Equation-of-motion orbital-optimized coupled-cluster 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.

Please note that Dunning's cc-pVXZ (X=2-6) basis sets [21, 22] 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,

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

0 H																	0 He
0 Li	0 Be											1 B	1 C	1 N	1 O	1 F	1 Ne
1 Na	1 Mg											5 Al	5 Si	5 P	5 S	5 Cl	5 Ar
5 K	5 Ca	5 Sc	5 Ti	5 V	5 Cr	5 Mn	5 Fe	5 Co	5 Ni	5 Cu	5 Zn	9 Ga	9 Ge	9 As	9 Se	9 Br	9 Kr
9 Rb	9 Sr	14 Y	14 Zr	14 Nb	14 Mo	14 Tc	14 Ru	14 Rh	14 Pd	14 Ag	14 Cd	18 In	18 Sn	18 Sb	18 Te	18 I	18 Xe
18 Cs	18 Ba	23 Lu	23 Hf	23 Ta	23 W	23 Re	23 Os	23 Ir	23 Pt	23 Au	23 Hg	34 Tl	34 Pb	34 Bi	34 Po	34 At	34 Rn
34 Fr	34 Ra	34 Lr	50 Rf	50 Db	50 Sg	50 Bh	50 Hs	50 Mt	50 Ds	50 Rg	50 Cn	50 Nh	50 Fl	50 Mc	50 Lv	50 Ts	50 Og

Lanthanides	18 La	18 Ce	18 Pr	18 Nd	18 Pm	18 Sm	18 Eu	18 Gd	18 Tb	18 Dy	18 Ho	18 Er	18 Tm	18 Yb
Actinides	34 Ac	34 Th	34 Pa	34 U	34 Np	34 Pu	34 Am	34 Cm	34 Bk	34 Cf	34 Es	34 Fm	34 Md	34 No

`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 R_A can be written as

$$G_{ijk}(a, \boldsymbol{r_A}) \;=\; \big(x - A_x\big)^i \big(y - A_y\big)^j \big(z - A_z\big)^k e^{-a\boldsymbol{r_A^2}}, \tag{1}$$

where i, j, k are the x, y, z components of the angular momentum, respectively, and $\boldsymbol{r}$ 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,

$$G_{lm}(\mathbf{r}, a, \mathbf{A}) \;=\; S_{lm}(x_A, y_A, z_A) e^{-a\boldsymbol{r_A^2}}, \tag{2}$$

where $S_{lm}(\mathbf{r_A})$ are the real solid-harmonics [20, 23]. The spherical GTOs can be obtained from the Cartesian GTOs by a simple transformation [23, 24].

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` [25] 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 5 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 4: Angular momentum convention used in `MacroQC`.

L	0	1	2	3	4	5	6	7	8	9	10
Orbital	s	p	d	f	g	h	i	k	l	m	n

Table 5: Built-In Basis Sets in `MacroQC`.

Basis Family	**Basis**	**JK Auxiliary Basis**	**MP2 (RI) Auxiliary Basis**
cc-pVDZ	cc-pvdz	cc-pvdz-jkfit	cc-pvdz-ri
cc-pV(D+d)Z	cc-pv_dpd_z	cc-pvdz-jkfit	cc-pvdz-ri
cc-pCVDZ	cc-pcvdz	-	-
cc-pCV(D+d)Z	cc-pcv_dpd_z	-	-
cc-pwCVDZ	cc-pwcvdz	cc-pvdz-jkfit	cc-pwcvdz-ri
cc-pwCV(D+d)Z	cc-pwcv_dpd_z	cc-pvdz-jkfit	cc-pwcvdz-ri
aug-cc-pVDZ	aug-cc-pvdz	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
aug-cc-pV(D+d)Z	aug-cc-pv_dpd_z	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
aug-cc-pCVDZ	aug-cc-pcvdz	-	-
aug-cc-pCV(D+d)Z	aug-cc-pcv_dpd_z	-	-
aug-cc-pwCVDZ	aug-cc-pwcvdz	aug-cc-pvdz-jkfit	aug-cc-pwcvdz-ri
aug-cc-pwCV(D+d)Z	aug-cc-pwcv_dpd_z	aug-cc-pvdz-jkfit	aug-cc-pwcvdz-ri
heavy-aug-cc-pVDZ	heavy-aug-cc-pvdz	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
heavy-aug-cc-pV(D+d)Z	heavy-aug-cc-pv_dpd_z	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
heavy-aug-cc-pCVDZ	heavy-aug-cc-pcvdz	-	-
heavy-aug-cc-pCV(D+d)Z	heavy-aug-cc-pcv_dpd_z	-	-
heavy-aug-cc-pwCVDZ	heavy-aug-cc-pwcvdz	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pwcvdz-ri
heavy-aug-cc-pwCV(D+d)Z	heavy-aug-cc-pwcv_dpd_z	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pwcvdz-ri
jun-cc-pVDZ	jun-cc-pvdz	jun-cc-pvdz-jkfit	jun-cc-pvdz-ri
jun-cc-pV(D+d)Z	jun-cc-pv_dpd_z	jun-cc-pvdz-jkfit	jun-cc-pvdz-ri
jun-cc-pCVDZ	jun-cc-pcvdz	-	-
jun-cc-pCV(D+d)Z	jun-cc-pcv_dpd_z	-	-
jun-cc-pwCVDZ	jun-cc-pwcvdz	jun-cc-pvdz-jkfit	jun-cc-pwcvdz-ri
jun-cc-pwCV(D+d)Z	jun-cc-pwcv_dpd_z	jun-cc-pvdz-jkfit	jun-cc-pwcvdz-ri
d-aug-cc-pVDZ	d-aug-cc-pvdz	-	-
d-aug-cc-pCVDZ	d-aug-cc-pcvdz	-	-
d-aug-cc-pwCVDZ	d-aug-cc-pwcvdz	-	-
cc-pVDZ-DK	cc-pvdz-dk	-	-
cc-pCVDZ-DK	cc-pcvdz-dk	-	-
aug-cc-pVDZ-DK	aug-cc-pvdz-dk	-	-
aug-cc-pCVDZ-DK	aug-cc-pcvdz-dk	-	-
heavy-aug-cc-pVDZ-DK	heavy-aug-cc-pvdz-dk	-	-
heavy-aug-cc-pCVDZ-DK	heavy-aug-cc-pcvdz-dk	-	-
cc-pVTZ	cc-pvtz	cc-pvtz-jkfit	cc-pvtz-ri
cc-pV(T+d)Z	cc-pv_tpd_z	cc-pvtz-jkfit	cc-pvtz-ri
cc-pCVTZ	cc-pcvtz	-	-
cc-pCV(T+d)Z	cc-pcv_tpd_z	-	-
cc-pwCVTZ	cc-pwcvtz	cc-pvtz-jkfit	cc-pwcvtz-ri
cc-pwCV(T+d)Z	cc-pwcv_tpd_z	cc-pvtz-jkfit	cc-pwcvtz-ri
aug-cc-pVTZ	aug-cc-pvtz	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
aug-cc-pV(T+d)Z	aug-cc-pv_tpd_z	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
aug-cc-pCVTZ	aug-cc-pcvtz	-	-
aug-cc-pCV(T+d)Z	aug-cc-pcv_tpd_z	-	-
aug-cc-pwCVTZ	aug-cc-pwcvtz	aug-cc-pvtz-jkfit	aug-cc-pwcvtz-ri
aug-cc-pwCV(T+d)Z	aug-cc-pwcv_tpd_z	aug-cc-pvtz-jkfit	aug-cc-pwcvtz-ri
heavy-aug-cc-pVTZ	heavy-aug-cc-pvtz	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
heavy-aug-cc-pV(T+d)Z	heavy-aug-cc-pv_tpd_z	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
heavy-aug-cc-pCVTZ	heavy-aug-cc-pcvtz	-	-
heavy-aug-cc-pCV(T+d)Z	heavy-aug-cc-pcv_tpd_z	-	-
heavy-aug-cc-pwCVTZ	heavy-aug-cc-pwcvtz	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pwcvtz-ri

heavy-aug-cc-pwCV(T+d)Z	heavy-aug-cc-pwcv_tpd_z	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pwcvtz-ri
jun-cc-pVTZ	jun-cc-pvtz	jun-cc-pvtz-jkfit	jun-cc-pvtz-ri
jun-cc-pV(T+d)Z	jun-cc-pv_tpd_z	jun-cc-pvtz-jkfit	jun-cc-pvtz-ri
jun-cc-pCVTZ	jun-cc-pcvtz	-	-
jun-cc-pCV(T+d)Z	jun-cc-pcv_tpd_z	-	-
jun-cc-pwCVTZ	jun-cc-pwcvtz	jun-cc-pvtz-jkfit	jun-cc-pwcvtz-ri
jun-cc-pwCV(T+d)Z	jun-cc-pwcv_tpd_z	jun-cc-pvtz-jkfit	jun-cc-pwcvtz-ri
may-cc-pVTZ	may-cc-pvtz	may-cc-pvtz-jkfit	may-cc-pvtz-ri
may-cc-pV(T+d)Z	may-cc-pv_tpd_z	may-cc-pvtz-jkfit	may-cc-pvtz-ri
may-cc-pCVTZ	may-cc-pcvtz	-	-
may-cc-pCV(T+d)Z	may-cc-pcv_tpd_z	-	-
may-cc-pwCVTZ	may-cc-pwcvtz	may-cc-pvtz-jkfit	may-cc-pwcvtz-ri
may-cc-pwCV(T+d)Z	may-cc-pwcv_tpd_z	may-cc-pvtz-jkfit	may-cc-pwcvtz-ri
d-aug-cc-pVTZ	d-aug-cc-pvtz	-	-
d-aug-cc-pCVTZ	d-aug-cc-pcvtz	-	-
d-aug-cc-pwCVTZ	d-aug-cc-pwcvtz	-	-
cc-pVTZ-DK	cc-pvtz-dk	-	-
cc-pCVTZ-DK	cc-pcvtz-dk	-	-
cc-pwCVTZ-DK	cc-pwcvtz-dk	-	-
aug-cc-pVTZ-DK	aug-cc-pvtz-dk	-	-
aug-cc-pCVTZ-DK	aug-cc-pcvtz-dk	-	-
aug-cc-pwCVTZ-DK	aug-cc-pwcvtz-dk	-	-
heavy-aug-cc-pVTZ-DK	heavy-aug-cc-pvtz-dk	-	-
heavy-aug-cc-pCVTZ-DK	heavy-aug-cc-pcvtz-dk	-	-
heavy-aug-cc-pwCVTZ-DK	heavy-aug-cc-pwcvtz-dk	-	-
cc-pVQZ	cc-pvqz	cc-pvqz-jkfit	cc-pvqz-ri
cc-pV(Q+d)Z	cc-pv_qpd_z	cc-pvqz-jkfit	cc-pvqz-ri
cc-pCVQZ	cc-pcvqz	-	-
cc-pCV(Q+d)Z	cc-pcv_qpd_z	-	-
cc-pwCVQZ	cc-pwcvqz	cc-pvqz-jkfit	cc-pwcvqz-ri
cc-pwCV(Q+d)Z	cc-pwcv_qpd_z	cc-pvqz-jkfit	cc-pwcvqz-ri
aug-cc-pVQZ	aug-cc-pvqz	aug-cc-pvqz-jkfit	aug-cc-pvqz-ri
aug-cc-pV(Q+d)Z	aug-cc-pv_qpd_z	aug-cc-pvqz-jkfit	aug-cc-pvqz-ri
aug-cc-pCVQZ	aug-cc-pcvqz	-	-
aug-cc-pCV(Q+d)Z	aug-cc-pcv_qpd_z	-	-
aug-cc-pwCVQZ	aug-cc-pwcvqz	aug-cc-pvqz-jkfit	aug-cc-pwcvqz-ri
aug-cc-pwCV(Q+d)Z	aug-cc-pwcv_qpd_z	aug-cc-pvqz-jkfit	aug-cc-pwcvqz-ri
heavy-aug-cc-pVQZ	heavy-aug-cc-pvqz	heavy-aug-cc-pvqz-jkfit	heavy-aug-cc-pvqz-ri
heavy-aug-cc-pV(Q+d)Z	heavy-aug-cc-pv_qpd_z	heavy-aug-cc-pvqz-jkfit	heavy-aug-cc-pvqz-ri
heavy-aug-cc-pCVQZ	heavy-aug-cc-pcvqz	-	-
heavy-aug-cc-pCV(Q+d)Z	heavy-aug-cc-pcv_qpd_z	-	-
heavy-aug-cc-pwCVQZ	heavy-aug-cc-pwcvqz	heavy-aug-cc-pvqz-jkfit	heavy-aug-cc-pwcvqz-ri
heavy-aug-cc-pwCV(Q+d)Z	heavy-aug-cc-pwcv_qpd_z	heavy-aug-cc-pvqz-jkfit	heavy-aug-cc-pwcvqz-ri
jun-cc-pVQZ	jun-cc-pvqz	jun-cc-pvqz-jkfit	jun-cc-pvqz-ri
jun-cc-pV(Q+d)Z	jun-cc-pv_qpd_z	jun-cc-pvqz-jkfit	jun-cc-pvqz-ri
jun-cc-pCVQZ	jun-cc-pcvqz	-	-
jun-cc-pCV(Q+d)Z	jun-cc-pcv_qpd_z	-	-
jun-cc-pwCVQZ	jun-cc-pwcvqz	jun-cc-pvqz-jkfit	jun-cc-pwcvqz-ri
jun-cc-pwCV(Q+d)Z	jun-cc-pwcv_qpd_z	jun-cc-pvqz-jkfit	jun-cc-pwcvqz-ri
may-cc-pVQZ	may-cc-pvqz	may-cc-pvqz-jkfit	may-cc-pvqz-ri
may-cc-pV(Q+d)Z	may-cc-pv_qpd_z	may-cc-pvqz-jkfit	may-cc-pvqz-ri
may-cc-pCVQZ	may-cc-pcvqz	-	-
may-cc-pCV(Q+d)Z	may-cc-pcv_qpd_z	-	-
may-cc-pwCVQZ	may-cc-pwcvqz	may-cc-pvqz-jkfit	may-cc-pwcvqz-ri

may-cc-pwCV(Q+d)Z	may-cc-pwcv_qpd_z	may-cc-pvqz-jkfit	may-cc-pwcvqz-ri
apr-cc-pVQZ	apr-cc-pvqz	apr-cc-pvqz-jkfit	apr-cc-pvqz-ri
apr-cc-pV(Q+d)Z	apr-cc-pv_qpd_z	apr-cc-pvqz-jkfit	apr-cc-pvqz-ri
apr-cc-pCVQZ	apr-cc-pcvqz	-	-
apr-cc-pCV(Q+d)Z	apr-cc-pcv_qpd_z	-	-
apr-cc-pwCVQZ	apr-cc-pwcvqz	apr-cc-pvqz-jkfit	apr-cc-pwcvqz-ri
apr-cc-pwCV(Q+d)Z	apr-cc-pwcv_qpd_z	apr-cc-pvqz-jkfit	apr-cc-pwcvqz-ri
d-aug-cc-pVQZ	d-aug-cc-pvqz	-	-
d-aug-cc-pCVQZ	d-aug-cc-pcvqz	-	-
d-aug-cc-pwCVQZ	d-aug-cc-pwcvqz	-	-
cc-pVQZ-DK	cc-pvqz-dk	-	-
cc-pCVQZ-DK	cc-pcvqz-dk	-	-
cc-pwCVQZ-DK	cc-pwcvqz-dk	-	-
aug-cc-pVQZ-DK	aug-cc-pvqz-dk	-	-
aug-cc-pCVQZ-DK	aug-cc-pcvqz-dk	-	-
aug-cc-pwCVQZ-DK	aug-cc-pwcvqz-dk	-	-
heavy-aug-cc-pVQZ-DK	heavy-aug-cc-pvqz-dk	-	-
heavy-aug-cc-pCVQZ-DK	heavy-aug-cc-pcvqz-dk	-	-
heavy-aug-cc-pwCVQZ-DK	heavy-aug-cc-pwcvqz-dk	-	-
cc-pV5Z	cc-pv5z	cc-pv5z-jkfit	cc-pv5z-ri
cc-pV(5+d)Z	cc-pv_5pd_z	cc-pv5z-jkfit	cc-pv5z-ri
cc-pCV5Z	cc-pcv5z	-	-
cc-pCV(5+d)Z	cc-pcv_5pd_z	-	-
cc-pwCV5Z	cc-pwcv5z	cc-pv5z-jkfit	cc-pwcv5z-ri
cc-pwCV(5+d)Z	cc-pwcv_5pd_z	cc-pv5z-jkfit	cc-pwcv5z-ri
aug-cc-pV5Z	aug-cc-pv5z	aug-cc-pv5z-jkfit	aug-cc-pv5z-ri
aug-cc-pV(5+d)Z	aug-cc-pv_5pd_z	aug-cc-pv5z-jkfit	aug-cc-pv5z-ri
aug-cc-pCV5Z	aug-cc-pcv5z	-	-
aug-cc-pCV(5+d)Z	aug-cc-pcv_5pd_z	-	-
aug-cc-pwCV5Z	aug-cc-pwcv5z	aug-cc-pv5z-jkfit	aug-cc-pwcv5z-ri
aug-cc-pwCV(5+d)Z	aug-cc-pwcv_5pd_z	aug-cc-pv5z-jkfit	aug-cc-pwcv5z-ri
heavy-aug-cc-pV5Z	heavy-aug-cc-pv5z	heavy-aug-cc-pv5z-jkfit	heavy-aug-cc-pv5z-ri
heavy-aug-cc-pV(5+d)Z	heavy-aug-cc-pv_5pd_z	heavy-aug-cc-pv5z-jkfit	heavy-aug-cc-pv5z-ri
heavy-aug-cc-pCV5Z	heavy-aug-cc-pcv5z	-	-
heavy-aug-cc-pCV(5+d)Z	heavy-aug-cc-pcv_5pd_z	-	-
heavy-aug-cc-pwCV5Z	heavy-aug-cc-pwcv5z	heavy-aug-cc-pv5z-jkfit	heavy-aug-cc-pwcv5z-ri
heavy-aug-cc-pwCV(5+d)Z	heavy-aug-cc-pwcv_5pd_z	heavy-aug-cc-pv5z-jkfit	heavy-aug-cc-pwcv5z-ri
jun-cc-pV5Z	jun-cc-pv5z	jun-cc-pv5z-jkfit	jun-cc-pv5z-ri
jun-cc-pV(5+d)Z	jun-cc-pv_5pd_z	jun-cc-pv5z-jkfit	jun-cc-pv5z-ri
jun-cc-pCV5Z	jun-cc-pcv5z	-	-
jun-cc-pCV(5+d)Z	jun-cc-pcv_5pd_z	-	-
jun-cc-pwCV5Z	jun-cc-pwcv5z	jun-cc-pv5z-jkfit	jun-cc-pwcv5z-ri
jun-cc-pwCV(5+d)Z	jun-cc-pwcv_5pd_z	jun-cc-pv5z-jkfit	jun-cc-pwcv5z-ri
may-cc-pV5Z	may-cc-pv5z	may-cc-pv5z-jkfit	may-cc-pv5z-ri
may-cc-pV(5+d)Z	may-cc-pv_5pd_z	may-cc-pv5z-jkfit	may-cc-pv5z-ri
may-cc-pCV5Z	may-cc-pcv5z	-	-
may-cc-pCV(5+d)Z	may-cc-pcv_5pd_z	-	-
may-cc-pwCV5Z	may-cc-pwcv5z	may-cc-pv5z-jkfit	may-cc-pwcv5z-ri
may-cc-pwCV(5+d)Z	may-cc-pwcv_5pd_z	may-cc-pv5z-jkfit	may-cc-pwcv5z-ri
apr-cc-pV5Z	apr-cc-pv5z	apr-cc-pv5z-jkfit	apr-cc-pv5z-ri
apr-cc-pV(5+d)Z	apr-cc-pv_5pd_z	apr-cc-pv5z-jkfit	apr-cc-pv5z-ri
apr-cc-pCV5Z	apr-cc-pcv5z	-	-
apr-cc-pCV(5+d)Z	apr-cc-pcv_5pd_z	-	-
apr-cc-pwCV5Z	apr-cc-pwcv5z	apr-cc-pv5z-jkfit	apr-cc-pwcv5z-ri

apr-cc-pwCV(5+d)Z	apr-cc-pwcv_5pd_z	apr-cc-pv5z-jkfit	apr-cc-pwcv5z-ri
mar-cc-pV5Z	mar-cc-pv5z	mar-cc-pv5z-jkfit	mar-cc-pv5z-ri
mar-cc-pV(5+d)Z	mar-cc-pv_5pd_z	mar-cc-pv5z-jkfit	mar-cc-pv5z-ri
mar-cc-pCV5Z	mar-cc-pcv5z	-	-
mar-cc-pCV(5+d)Z	mar-cc-pcv_5pd_z	-	-
mar-cc-pwCV5Z	mar-cc-pwcv5z	mar-cc-pv5z-jkfit	mar-cc-pwcv5z-ri
mar-cc-pwCV(5+d)Z	mar-cc-pwcv_5pd_z	mar-cc-pv5z-jkfit	mar-cc-pwcv5z-ri
d-aug-cc-pV5Z	d-aug-cc-pv5z	-	-
d-aug-cc-pCV5Z	d-aug-cc-pcv5z	-	-
d-aug-cc-pwCV5Z	d-aug-cc-pwcv5z	-	-
cc-pV5Z-DK	cc-pv5z-dk	-	-
cc-pCV5Z-DK	cc-pcv5z-dk	-	-
cc-pwCV5Z-DK	cc-pwcv5z-dk	-	-
aug-cc-pV5Z-DK	aug-cc-pv5z-dk	-	-
aug-cc-pCV5Z-DK	aug-cc-pcv5z-dk	-	-
aug-cc-pwCV5Z-DK	aug-cc-pwcv5z-dk	-	-
heavy-aug-cc-pV5Z-DK	heavy-aug-cc-pv5z-dk	-	-
heavy-aug-cc-pCV5Z-DK	heavy-aug-cc-pcv5z-dk	-	-
heavy-aug-cc-pwCV5Z-DK	heavy-aug-cc-pwcv5z-dk	-	-
cc-pV6Z	cc-pv6z	-	cc-pv6z-ri
cc-pV(6+d)Z	cc-pv_6pd_z	-	cc-pv6z-ri
cc-pCV6Z	cc-pcv6z	-	-
cc-pCV(6+d)Z	cc-pcv_6pd_z	-	-
aug-cc-pV6Z	aug-cc-pv6z	-	aug-cc-pv6z-ri
aug-cc-pV(6+d)Z	aug-cc-pv_6pd_z	-	aug-cc-pv6z-ri
aug-cc-pCV6Z	aug-cc-pcv6z	-	-
aug-cc-pCV(6+d)Z	aug-cc-pcv_6pd_z	-	-
heavy-aug-cc-pV6Z	heavy-aug-cc-pv6z	-	heavy-aug-cc-pv6z-ri
heavy-aug-cc-pV(6+d)Z	heavy-aug-cc-pv_6pd_z	-	heavy-aug-cc-pv6z-ri
heavy-aug-cc-pCV6Z	heavy-aug-cc-pcv6z	-	-
heavy-aug-cc-pCV(6+d)Z	heavy-aug-cc-pcv_6pd_z	-	-
jun-cc-pV6Z	jun-cc-pv6z	-	jun-cc-pv6z-ri
jun-cc-pV(6+d)Z	jun-cc-pv_6pd_z	-	jun-cc-pv6z-ri
jun-cc-pCV6Z	jun-cc-pcv6z	-	-
jun-cc-pCV(6+d)Z	jun-cc-pcv_6pd_z	-	-
may-cc-pV6Z	may-cc-pv6z	-	may-cc-pv6z-ri
may-cc-pV(6+d)Z	may-cc-pv_6pd_z	-	may-cc-pv6z-ri
may-cc-pCV6Z	may-cc-pcv6z	-	-
may-cc-pCV(6+d)Z	may-cc-pcv_6pd_z	-	-
apr-cc-pV6Z	apr-cc-pv6z	-	apr-cc-pv6z-ri
apr-cc-pV(6+d)Z	apr-cc-pv_6pd_z	-	apr-cc-pv6z-ri
apr-cc-pCV6Z	apr-cc-pcv6z	-	-
apr-cc-pCV(6+d)Z	apr-cc-pcv_6pd_z	-	-
mar-cc-pV6Z	mar-cc-pv6z	-	mar-cc-pv6z-ri
mar-cc-pV(6+d)Z	mar-cc-pv_6pd_z	-	mar-cc-pv6z-ri
mar-cc-pCV6Z	mar-cc-pcv6z	-	-
mar-cc-pCV(6+d)Z	mar-cc-pcv_6pd_z	-	-
feb-cc-pV6Z	feb-cc-pv6z	-	feb-cc-pv6z-ri
feb-cc-pV(6+d)Z	feb-cc-pv_6pd_z	-	feb-cc-pv6z-ri
feb-cc-pCV6Z	feb-cc-pcv6z	-	-
feb-cc-pCV(6+d)Z	feb-cc-pcv_6pd_z	-	-
d-aug-cc-pV6Z	d-aug-cc-pv6z	-	-
d-aug-cc-pCV6Z	d-aug-cc-pcv6z	-	-
STO-3G	sto-3g	def2-universal-jkfit	def2-svp-ri

STO-6G	sto-6g	def2-universal-jkfit	def2-svp-ri
3-21G	3-21g	def2-universal-jkfit	def2-svp-ri
6-31G	6-31g	cc-pvdz-jkfit	cc-pvdz-ri
6-31G(d)	6-31g_d_	cc-pvdz-jkfit	cc-pvdz-ri
6-31G(d,p)	6-31g_d_p_	cc-pvdz-jkfit	cc-pvdz-ri
6-31G*	6-31g_d_	cc-pvdz-jkfit	cc-pvdz-ri
6-31G**	6-31g_d_p_	cc-pvdz-jkfit	cc-pvdz-ri
6-31+G	6-31pg	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
6-31+G(d)	6-31pg_d_	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
6-31+G(d,p)	6-31pg_d_p_	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
6-31+G*	6-31pg_d_	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
6-31+G**	6-31pg_d_p_	heavy-aug-cc-pvdz-jkfit	heavy-aug-cc-pvdz-ri
6-31++G	6-31ppg	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
6-31++G(d)	6-31ppg_d_	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
6-31++G(d,p)	6-31ppg_d_p_	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
6-31++G*	6-31ppg_d_	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
6-31++G**	6-31ppg_d_p_	aug-cc-pvdz-jkfit	aug-cc-pvdz-ri
6-311G	6-311g	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(d)	6-311g_d_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(d,p)	6-311g_d_p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G*	6-311g_d_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G**	6-311g_d_p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2d)	6-311g_2d_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2d,p)	6-311g_2d_p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2d,2p)	6-311g_2d_2p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2df)	6-311g_2df_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2df,p)	6-311g_2df_p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2df,2p)	6-311g_2df_2p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(2df,2pd)	6-311g_2df_2pd_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(3df)	6-311g_3df_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(3df,p)	6-311g_3df_p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(3df,2p)	6-311g_3df_2p_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(3df,2pd)	6-311g_3df_2pd_	cc-pvtz-jkfit	cc-pvtz-ri
6-311G(3df,3pd)	6-311g_3df_3pd_	cc-pvtz-jkfit	cc-pvtz-ri
6-311+G	6-311pg	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(d)	6-311pg_d_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(d,p)	6-311pg_d_p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G*	6-311pg_d_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G**	6-311pg_d_p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2d)	6-311pg_2d_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2d,p)	6-311pg_2d_p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2d,2p)	6-311pg_2d_2p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2df)	6-311pg_2df_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2df,p)	6-311pg_2df_p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2df,2p)	6-311pg_2df_2p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(2df,2pd)	6-311pg_2df_2pd_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(3df)	6-311pg_3df_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(3df,p)	6-311pg_3df_p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(3df,2p)	6-311pg_3df_2p_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(3df,2pd)	6-311pg_3df_2pd_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311+G(3df,3pd)	6-311pg_3df_3pd_	heavy-aug-cc-pvtz-jkfit	heavy-aug-cc-pvtz-ri
6-311++G	6-311ppg	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(d)	6-311ppg_d_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(d,p)	6-311ppg_d_p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri

6-311++G*	6-311ppg_d_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G**	6-311ppg_d_p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2d)	6-311ppg_2d_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2d,p)	6-311ppg_2d_p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2d,2p)	6-311ppg_2d_2p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2df)	6-311ppg_2df_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2df,p)	6-311ppg_2df_p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2df,2p)	6-311ppg_2df_2p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(2df,2pd)	6-311ppg_2df_2pd_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(3df)	6-311ppg_3df_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(3df,p)	6-311ppg_3df_p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(3df,2p)	6-311ppg_3df_2p_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(3df,2pd)	6-311ppg_3df_2pd_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
6-311++G(3df,3pd)	6-311ppg_3df_3pd_	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
def2-SV(P)	def2-sv_p_	def2-universal-jkfit	def2-sv_p_-ri
def2-mSVP	def2-msvp	def2-universal-jkfit	def2-svp-ri
def2-SVP	def2-svp	def2-universal-jkfit	def2-svp-ri
def2-SVPD	def2-svpd	def2-universal-jkfit	def2-svpd-ri
def2-TZVP	def2-tzvp	def2-universal-jkfit	def2-tzvp-ri
def2-TZVPD	def2-tzvpd	def2-universal-jkfit	def2-tzvpd-ri
def2-TZVPP	def2-tzvpp	def2-universal-jkfit	def2-tzvpp-ri
def2-TZVPPD	def2-tzvppd	def2-universal-jkfit	def2-tzvppd-ri
def2-QZVP	def2-qzvp	def2-universal-jkfit	def2-qzvp-ri
def2-QZVPD	def2-qzvpd	def2-universal-jkfit	-
def2-QZVPP	def2-qzvpp	def2-universal-jkfit	def2-qzvpp-ri
def2-QZVPPD	def2-qzvppd	def2-universal-jkfit	def2-qzvppd-ri
aug-pcseg-0	aug-pcseg-0	def2-universal-jkfit	def2-sv_p_-ri
aug-pcseg-1	aug-pcseg-1	def2-universal-jkfit	def2-svpd-ri
aug-pcseg-2	aug-pcseg-2	def2-universal-jkfit	def2-tzvppd-ri
aug-pcseg-3	aug-pcseg-3	def2-universal-jkfit	def2-qzvppd-ri
aug-pcseg-4	aug-pcseg-4	aug-cc-pv5z-jkfit	aug-cc-pv5z-ri
aug-pcSseg-0	aug-pcsseg-0	def2-universal-jkfit	def2-sv_p_-ri
aug-pcSseg-1	aug-pcsseg-1	def2-universal-jkfit	def2-svpd-ri
aug-pcSseg-2	aug-pcsseg-2	def2-universal-jkfit	def2-tzvppd-ri
aug-pcSseg-3	aug-pcsseg-3	def2-universal-jkfit	def2-qzvppd-ri
aug-pcSseg-4	aug-pcsseg-4	aug-cc-pv5z-jkfit	aug-cc-pwcv5z-ri
pcseg-0	pcseg-0	def2-universal-jkfit	def2-sv_p_-ri
pcseg-1	pcseg-1	def2-universal-jkfit	def2-svp-ri
pcseg-2	pcseg-2	def2-universal-jkfit	def2-tzvpp-ri
pcseg-3	pcseg-3	def2-universal-jkfit	def2-qzvpp-ri
pcseg-4	pcseg-4	cc-pv5z-jkfit	cc-pv5z-ri
pcSseg-0	pcsseg-0	def2-universal-jkfit	def2-sv_p_-ri
pcSseg-1	pcsseg-1	def2-universal-jkfit	def2-svp-ri
pcSseg-2	pcsseg-2	def2-universal-jkfit	def2-tzvpp-ri
pcSseg-3	pcsseg-3	def2-universal-jkfit	def2-qzvpp-ri
pcSseg-4	pcsseg-4	cc-pv5z-jkfit	cc-pwcv5z-ri
minix	minix	def2-universal-jkfit	def2-svp-ri
DZ	dz	-	-
DZP	dzp	-	-
DZVP	dzvp	dgauss-dzvp-mix	dgauss-dzvp-autoaux
psi3-DZP	psi3-dzp	-	-
psi3-TZ2P	psi3-tz2p	-	-
psi3-TZ2PF	psi3-tz2pf	-	-
sadlej-lpol-dl	sadlej-lpol-dl	-	-

sadlej-lpol-ds	sadlej-lpol-ds	-	-
sadlej-lpol-fl	sadlej-lpol-fl	-	-
sadlej-lpol-fs	sadlej-lpol-fs	-	-
2zapa-nr	2zapa-nr	aug-cc-pvtz-jkfit	aug-cc-pvtz-ri
3zapa-nr	3zapa-nr	aug-cc-pvqz-jkfit	aug-cc-pvqz-ri
4zapa-nr	4zapa-nr	aug-cc-pv5z-jkfit	aug-cc-pv5z-ri
5zapa-nr	5zapa-nr	aug-cc-pv5z-jkfit	aug-cc-pv6z-ri
6zapa-nr	6zapa-nr	aug-cc-pv6z-ri	aug-cc-pv6z-ri
7zapa-nr	7zapa-nr	aug-cc-pv6z-ri	aug-cc-pv6z-ri
cc-pvqz-f12	cc-pvqz-f12	cc-pv5z-jkfit	cc-pv5z-ri
cc-pvtz-f12	cc-pvtz-f12	cc-pvqz-jkfit	cc-pvqz-ri
cc-pvqz-f12	cc-pvqz-f12	cc-pv5z-jkfit	cc-pv5z-ri

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:

```
$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...
 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...
 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.
```

⋮

```
 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...
 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-RI aux basis set file will be read for the C atom...
CC-PVDZ-RI aux basis set file will be read for the C atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
CC-PVDZ-RI aux basis set file will be read for the H atom...
DEF2-QZVP-RI aux basis set file will be read for the ZN atom...
```

7.4 User-Defined Basis Sets

`MacroQC` uses Gaussian format for basis sets. For example, the `cc-pvdz` basis set for the H atom:

```
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
****
```

To submit a general use-defined basis to `MacroQC` one needs to prepare a `.gbs` file in which the basis set should be given in the Gaussian format. For example, if one prepares a basis set file titled `mybasis.gbs`, this file should be placed in the `MQCIDIR/macroqc/share/basis` directory. `MQCIDIR` is the directory where `MacroQC` is installed. For example: `/opt/macroqc`. Many basis sets can be obtained from the *Basis Set Exchange* [26] website: Basis Set Exchange

7.5 Integral Library Options

- **PRINT_LEVEL**

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

- **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: Sring
  DEFAULT: AUTO
  OPTIONS: AUTO, OS1, OS2, MD4

- **BOYS_METHOD**
  The algorithm is used to compute grid values.
  TYPE: Sring
  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) [27], and the quadratically convergent SCF algorithm (QC-SCF). The `scf` module employs several initial guesses, such as the generalized Wolfsberg-Helmholtz (GWH) [28], 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 [20].

8.2 ROHF

The ROHF Fock matrix can be written as follows:

$$\boldsymbol{F} = \begin{pmatrix} F_{cc} & F_{co} & F_{cv} \\ F_{oc} & F_{oo} & F_{ov} \\ F_{vc} & F_{vo} & F_{vv} \end{pmatrix} \tag{4}$$

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

$$\boldsymbol{F} = \begin{bmatrix} A_{cc}F^{\alpha} + B_{cc}F^{\beta} & F^{\beta} & \frac{1}{2}\left(F^{\alpha} + F^{\beta}\right) \\ F^{\beta} & A_{oo}F^{\alpha} + B_{oo}F^{\beta} & F^{\alpha} \\ \frac{1}{2}\left(F^{\alpha} + F^{\beta}\right) & F^{\alpha} & A_{vv}F^{\alpha} + B_{vv}F^{\beta} \end{bmatrix} \tag{5}$$

In `MacroQC` there are several options available for ROHF coefficients (Table 6), such as Guest and Saunders [29], Roothan [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].

Table 6: 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	1	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
    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
$end_globals
```

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

```
$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
    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
  OPTIONS: TRUE, 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
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **SAD_DIIS**
  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: SAD
  OPTIONS: SAD, GWH, 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

- **RETRY**
  If SCF iterations are not converged, MacroQC tries different SCF algorithms and initial guesses.
  TYPE: BOOL
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

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:

```
$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 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 after every frequency computation).

```
$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 opt
$end_globals

$set_opt
   read_cart_hess true
   opt_algorithm qn
   opt_maxiter 50
   tol_rmsG 3.0E-4
   tol_maxG 4.5E-4
   tol_rmsDisp 1.2E-3
   tol_maxDisp 1.8E-3
$end_opt
```

9.3 OPT Module Options

- **OPT_TYPE**
  Type of the optimization.
  TYPE: Sring
  DEFAULT: MIN
  OPTIONS: MIN, TS
  - MIN : Minimum search
  - TS : Transition-state search
- **OPT_ALGORITHM**
  The geometry optimization algorithm.
  TYPE: Sring
  DEFAULT: QN
  OPTIONS: QN, RFO, SD
  - QN : Quasi-Newton
  - RFO : Rational Function Optimization
  - SD : Steepest Descent
- **HESS_UPDATE**
  The hessian update algorithm.
  TYPE: Sring
  DEFAULT: BFGS
  OPTIONS: BFGS, BOFILL
  - BFGS : Broyden-Fletcher-Goldfarb-Shannoa 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 optmization: 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?
  TYPE: Boolean
  DEFAULT: TRUE
  OPTIONS: TRUE, FALSE

- **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

- **OPT_COORD**
  The coordinate system that will be used in optimization.
  TYPE: Sring
  DEFAULT: RIC
  OPTIONS: RIC, CART

  - RIC : Redundant internal coordinates.
  - CART : Cartesian coordinates.

- **TOL_RMSDX**
  Convergence criterion for redundant internal coordinates to cartesian coordinates transformation iterations: $rms(\Delta X)$
  TYPE: Double

DEFAULT: 1.0e-5
OPTIONS: n User-defined value.

- **TOL_MAXDX**
  Convergence criterion for redundant internal coordinates to cartesian coordinates transformation iterations: $max(\Delta X)$
  TYPE: Double
  DEFAULT: 1.0e-4
  OPTIONS: n User-defined value.

- **RIC2CART_MAXITER**
  The maximum number of redundant internal coordiantes to cartesian coordinates transformation steps.
  TYPE: Integer
  DEFAULT: 50
  OPTIONS: n User-defined value.

- **MAX_STEP_X**
  Maximum redundant internal coordiantes to cartesian coordinates transformation step size in Bohr.
  TYPE: Double
  DEFAULT: 0.01
  OPTIONS: n User-defined value.

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 [41], Qchem [42], Cfour [43], Molpro [44], and Orca [45].

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
```

```
$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method mp2
  reference rhf
  jobtype freq
  freeze_core true
  step_bohr 0.05
$end_globals

$set_freq
  anharm_level 0
  derive_level 2
  mode serial
  temperature 298.15
  pressure 101325.0
  point_group C1
  axis_n 1
  energy -76.22842455
$end_freq
```

*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.

```
$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 mp2
  reference rhf
  jobtype freq
  freeze_core true
  step_bohr 0.05
$end_globals

$set_freq
```

```
  anharm_level 2
  derive_level 4
  mode sow
$end_freq
```

*When each gradient computation is completed, you can proceed to the frequency computation by selecting `MODE REAP`:

```
$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
  jobtype freq
  step_bohr 0.05
$end_globals

$set_freq
  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
    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
  jobtype freq
  step_bohr 0.05
```

```
$end_globals

$set_freq
  anharm_level 2
  derive_level 4
  mode sow
  software psi4
$end_freq

$SET_DG_EXTERNAL
set {
basis cc-pvdz
df_basis_scf cc-pvdz-jkfit
df_basis_cc cc-pvdz-ri
reference rhf
scf_type df
mp_type df
freeze_core true
qc_module occ
oeprop true
}
gradient('mp2')
$END_DG_EXTERNAL
```

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 correttion (cubic terms).
  - 2: Anharmonic correttion (quartic terms).

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

- **TOL_ANHARM**
  Tolerance value for Fermi resonance [46].
  TYPE: Double
  DEFAULT: 200.0
  OPTIONS: n User-defined value.

- **TOL_MARTIN**
  Tolerance value for criterion suggested by Martin and co-workers [47].
  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 Moller–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 Moller–Plesset perturbation theory as implemented in MACROQC.

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

$$\widetilde{\hat{p}}^{\dagger} = e^{\hat{K}} \hat{p}^{\dagger} e^{-\hat{K}} \tag{6}$$

$$\widetilde{\hat{p}} = e^{\hat{K}}\ \hat{p}\ e^{-\hat{K}} \tag{7}$$

$$|\widetilde{p}\rangle = e^{\hat{K}}\ |p\rangle \tag{8}$$

where $\hat{K}$ is the orbital rotation operator

$$\hat{K} = \sum_{p,q} K_{pq}\ \hat{E}_{pq} = \sum_{p>q} \kappa_{pq}\ \hat{E}^{-}_{pq} \tag{9}$$

$$\hat{E}_{pq} = \hat{p}^{\dagger}\hat{q} \tag{10}$$

$$\hat{E}^{-}_{pq} = \hat{E}_{pq}\ -\ \hat{E}_{qp} \tag{11}$$

$$\mathbf{K} = Skew(\kappa) \tag{12}$$

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

$$\mathbf{C}(\kappa) = \mathbf{C}^{(\mathbf{0})}\ e^{\mathbf{K}} \tag{13}$$

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

OMP2 [48, 49, 11, 10, 12]:

$$\begin{aligned}\widetilde{E}(\boldsymbol{\kappa}) &= \langle 0|\hat{H}^{\kappa}|0\rangle\ +\ \langle 0|\big(\hat{W}^{\kappa}_{N}\hat{T}^{(1)}_{2}\big)_{c}|0\rangle \\ &+\ \langle 0|\big\{\hat{\Lambda}^{(1)}_{2}\ \big(\hat{f}^{\kappa}_{N}\hat{T}^{(1)}_{2}\ +\ \hat{W}^{\kappa}_{N}\big)_{c}\big\}_{c}|0\rangle,\end{aligned} \tag{14}$$

OMP3 [50, 51, 52, 13, 18]:

$$\begin{aligned}\widetilde{E}(\boldsymbol{\kappa}) &= \langle 0|\hat{H}^{\kappa}|0\rangle \\ &+\ \langle 0|\big(\hat{W}^{\kappa}_{N}\hat{T}^{(1)}_{2}\big)_{c}|0\rangle\ +\ \langle 0|\big(\hat{W}^{\kappa}_{N}\hat{T}^{(2)}_{2}\big)_{c}|0\rangle \\ &+\ \langle 0|\{\hat{\Lambda}^{(1)}_{2}\ \big(\hat{f}^{\kappa}_{N}\hat{T}^{(1)}_{2}\ +\ \hat{W}^{\kappa}_{N}\big)_{c}\}_{c}|0\rangle \\ &+\ \langle 0|\{\hat{\Lambda}^{(1)}_{2}\ \big(\hat{f}^{\kappa}_{N}\hat{T}^{(2)}_{2}\ +\ \hat{W}^{\kappa}_{N}\hat{T}^{(1)}_{2}\big)_{c}\}_{c}|0\rangle \\ &+\ \langle 0|\{\hat{\Lambda}^{(2)}_{2}\ \big(\hat{f}^{\kappa}_{N}\hat{T}^{(1)}_{2}\ +\ \hat{W}^{\kappa}_{N}\big)_{c}\}_{c}|0\rangle,\end{aligned} \tag{15}$$

OMP2.5 [53, 54, 13]:

$$\begin{aligned}\widetilde{E}(\boldsymbol{\kappa}) &= \langle 0|\hat{H}|0\rangle \\ &+ \langle 0|\big(\hat{W}_N^\kappa \hat{T}_2^{(1)}\big)_c|0\rangle \ + \ \frac{1}{2}\langle 0|\big(\hat{W}_N^\kappa \hat{T}_2^{(2)}\big)_c|0\rangle \\ &+ \langle 0|\{\hat{\Lambda}_2^{(1)}\ \big(\hat{f}_N^\kappa \hat{T}_2^{(1)} \ + \ \hat{W}_N^\kappa\big)_c\}_c|0\rangle \\ &+ \frac{1}{2}\langle 0|\{\hat{\Lambda}_2^{(1)}\ \big(\hat{f}_N^\kappa \hat{T}_2^{(2)} \ + \ \hat{W}_N^\kappa \hat{T}_2^{(1)}\big)_c\}_c|0\rangle \\ &+ \frac{1}{2}\langle 0|\{\hat{\Lambda}_2^{(2)}\ \big(\hat{f}_N^\kappa \hat{T}_2^{(1)} \ + \ \hat{W}_N^\kappa\big)_c\}_c|0\rangle, \end{aligned} \tag{16}$$

OLCCD [55, 56, 14]:

$$\begin{aligned}\widetilde{E}(\boldsymbol{\kappa}) &= \langle 0|\hat{H}^\kappa|0\rangle \ + \ \langle 0|\big(\hat{W}_N^\kappa \hat{T}_2\big)_c|0\rangle \\ &+ \langle 0|\{\hat{\Lambda}_2\ \big(\hat{W}_N^\kappa \ + \ \hat{H}_N^\kappa \hat{T}_2\big)_c\}_c|0\rangle, \end{aligned} \tag{17}$$

OCCD [48, 57]:

$$\widetilde{E}(\boldsymbol{\kappa}) = \langle 0|(1+\hat{\Lambda}_2)e^{-\hat{T}_2}\hat{H}^\kappa e^{\hat{T}_2}|0\rangle, \tag{18}$$

where subscript c means only connected diagrams are included, $\hat{T}_2$ and $\hat{\Lambda}_2$ are cluster double excitation and de-excitation operators, respectfully, and $\hat{H}^\kappa$, $\hat{f}_N^\kappa$, $\hat{W}_N^\kappa$, and $\hat{H}_N^\kappa$ are defined as

$$\hat{H}^\kappa = e^{-\hat{K}}\hat{H}e^{\hat{K}}, \tag{19}$$
$$\hat{f}_N^\kappa = e^{-\hat{K}}\hat{f}_N^d e^{\hat{K}}, \tag{20}$$
$$\hat{W}_N^\kappa = e^{-\hat{K}}\hat{W}_N e^{\hat{K}}, \tag{21}$$
$$\hat{H}_N^\kappa = e^{-\hat{K}}\hat{H}_N e^{\hat{K}}. \tag{22}$$

The energy gradient and Hessian can be written as follows:

$$w_{pq} = \left.\frac{\partial \widetilde{E}}{\partial \kappa_{pq}}\right|_{\boldsymbol{\kappa}=0}, \tag{23}$$

$$A_{pq,rs} = \left.\frac{\partial^2 \widetilde{E}}{\partial \kappa_{pq}\partial \kappa_{rs}}\right|_{\boldsymbol{\kappa}=0}, \tag{24}$$

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

$$\widetilde{E}^{(2)}(\boldsymbol{\kappa}) = \widetilde{E}^{(0)} + \boldsymbol{\kappa}^\dagger w + \frac{1}{2}\,\boldsymbol{\kappa}^\dagger A\kappa, \tag{25}$$

where $\boldsymbol{\kappa}$ is the MO rotation vector, $\boldsymbol{w}$ is the MO gradient vector, and $\boldsymbol{A}$ is the MO Hessian matrix. Hence, minimizing the energy with respect to $\boldsymbol{\kappa}$ yields

$$\boldsymbol{\kappa} = -\boldsymbol{A}^{-1}w. \tag{26}$$

Table 7: OO methods.

Name	Calls Method	Energy	Gradient
`omp2`	Density-Fitted Orbital-Optimized MP2	RHF/UHF/ROHF	RHF/UHF/ROHF
`omp3`	Density-Fitted Orbital-Optimized MP3	RHF/UHF/ROHF	RHF/UHF/ROHF
`omp2.5`	Density-Fitted Orbital-Optimized MP2.5	RHF/UHF/ROHF	RHF/UHF/ROHF
`olccd`	Density-Fitted Orbital-Optimized LCCD	RHF/UHF/ROHF	RHF/UHF/ROHF
`occd`	Density-Fitted Orbital-Optimized CCD	RHF/UHF/ROHF	RHF/UHF/ROHF
`occd(t)`	Density-Fitted Orbital-Optimized CCD(T)	RHF/UHF/ROHF	NA
`occd(at)`	Density-Fitted Orbital-Optimized CCD(AT)	RHF/UHF/ROHF	NA

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 [48, 49, 11, 10, 12].

OMP3 Bozkaya and co-workers [50, 51, 52, 13, 18].

OMP2.5 Bozkaya and co-workers [53, 54, 13].

OLCCD Bozkaya and co-workers [55, 56, 14].

OCCD Bozkaya and co-workers [57].

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.

Table 8: Non-OO methods.

Name	Calls Method	Energy	Gradient
`mp2`	Density-Fitted MP2	RHF/UHF	RHF/UHF
`mp3`	Density-Fitted MP3	RHF/UHF	RHF/UHF
`mp2.5`	Density-Fitted MP2.5	RHF/UHF	RHF/UHF
`lccd`	Density-Fitted LCCD	RHF/UHF	RHF/UHF
`ccd`	Density-Fitted CCD	RHF/UHF	RHF/UHF
`ccsd`	Density-Fitted CCSD	RHF/UHF	RHF/UHF
`ccsd(t)`	Density-Fitted CCSD(T)	RHF/UHF	RHF/UHF
`ccsd(at)`	Density-Fitted CCSD(AT)	RHF/UHF	NA

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

MP2 Bozkaya and co-workers [48, 49, 11, 10, 12].

MP3 Bozkaya and co-workers [50, 51, 52, 13, 18].

MP2.5 Bozkaya and co-workers [53, 54, 13].

LCCD Bozkaya and co-workers [55, 56, 14].

LCCD Bozkaya and co-workers [55, 56, 14].

CCD Bozkaya and co-workers [48, 15, 16, 57].

CCSD Bozkaya and co-workers [48, 15, 16, 57].

CCSD(T) Bozkaya and co-workers [58, 15, 17].

CCSD(AT) Bozkaya and co-workers [58, 15].

11.3 Input Examples

*Single point `omp2/cc-pvdz` computation:

```
$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 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:

```
$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
    e_convergence 1e-8
    r_convergence 1e-6
$end_dfocc
```

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

```
$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
    e_convergence 1e-8
    r_convergence 1e-6
    nat_orbs true
    occ_tolerance 1.0e-5
$end_dfocc
```

*Ionization potentials via the extended Koopmans' theorem (EKT) at the `ccsd(t)/cc-pvdz` level:

```
$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:

```
$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
```

```
    e_convergence 1e-8
    r_convergence 1e-6
    nat_orbs true
    occ_tolerance 1e-5
$end_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**
  Scaling value for 3rd order energy correction (S. Grimme, Vol. 24, pp. 1529, J. Comput. Chem.)
  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 $\langle \hat{S}^2 \rangle$ for DF-OMP2/DF-MP2?
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE.

- **DIAG_METHOD**
  Diagonalization method used for the CIS Hamiltonian.

TYPE: String
DEFAULT: DAVIDSON
OPTIONS: DAVIDSON, FULL_DIAG

- 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???**
  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?

TYPE: Boolean
DEFAULT: FALSE
OPTIONS: TRUE, FALSE

- **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, POPLE

- **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

- **EOM_MP2**
  Do compute eom_mp2 intermediates for EOM-MP2 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 [19].

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

$set_globals
    basis cc-pvdz
    method qdpt2
    reference uhf
    jobtype energy
    freeze_core false
$end_globals

$set_qdpt
    qdpt_maxiter 50
    nactmo 2
    nalpha 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
$end_molecule

$set_globals
    basis cc-pvdz
    method cas-ci
    reference uhf
    jobtype energy
    freeze_core false
$end_globals

$set_qdpt
    qdpt_maxiter 50
    nactmo 2
    nalpha 1
    nbeta 1
$end_qdpt
```

*Single point `fci/cc-pvdz` computation:

```
$set_molecule H2O
    0 1
    B   0.00000   0.00000   0.00000
    H   0.00000   0.00000   1.00000
$end_molecule

$set_globals
    basis cc-pvdz
    method fci
    reference uhf
    jobtype energy
    freeze_core false
    memory 6000
$end_globals

$set_qdpt
    qdpt_maxiter 50
$end_qdpt
```

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 a 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 [59, 60].

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
```

```
$set_globals
  basis cc-pvdz
  aux_basis_scf cc-pvdz-jkfit
  aux_basis_corr cc-pvdz-ri
  method eom-ccsd
  reference rhf
  jobtype energy
  freeze_core true
$end_globals

$set_eomee
  num_roots 5
$end_eomee
```

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.
  TYPE: Double
  DEFAULT: 1.0e-6
  OPTIONS: n User-defined tolerance value.

- **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 $c_i^a = -1/\sqrt{2}$.

- **SYMMETRIC_HEFF**
  Symmetrize the effective Hamiltonian.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

- **PPL_TYPE**
  Type of the EOM-CCSD PPL term.
  TYPE: String
  DEFAULT: AUTO
  OPTIONS: AUTO, CD

- **CD_PRESCREEN**
  Do apply prescreening for the CD algorithm? This option can be used when the ppl_type is cd.
  TYPE: Boolean
  DEFAULT: FALSE
  OPTIONS: TRUE, FALSE

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 [61] 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     -1.643066000
H       -3.246081000      0.370694000     -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
```

```
H         2.534735000      -1.691722000       0.201172000
H         2.682886000      -0.885237000      -1.360734000
H         4.865160000      -0.810408000      -0.118293000
H         4.233578000       0.839788000      -0.330129000
H         4.083420000       0.026781000       1.243777000
$end_molecule

$set_globals
    basis cc-pvdz
    aux_basis_scf cc-pvdz-jkfit
    method scf
    reference rhf
    jobtype energy
    mol_frag true
    memory 1000
$end_globals

$set_lssmf
    level 3
    nb_cutoff_max 10.0
    mode serial
$end_lssmf
```

In this example, `mol_frag true` option request 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 Δ_{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 splitted into 0 parts
Bonded files will be splitted 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, Δ_{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.

```
************************************************************
                        SCF ENERGY
************************************************************

Coeff |          Energy |    Coeff*Energy | Bonded Group Name |
------|-----------------|-----------------|-------------------|
  1   | -157.3064946038 | -157.3064946038 |         G_1_2_3_4 |
  1   | -157.3055496928 | -157.3055496928 |         G_1_2_3_5 |
  1   | -157.3038834931 | -157.3038834931 |         G_1_2_3_6 |
  1   | -157.3061863028 | -157.3061863028 |         G_2_3_4_5 |
  1   | -157.3049150998 | -157.3049150998 |         G_2_3_4_6 |
  1   | -157.3052965489 | -157.3052965489 |         G_2_3_5_6 |
  1   | -157.3050898279 | -157.3050898279 |         G_2_3_6_7 |
  1   | -157.3069522452 | -157.3069522452 |         G_3_4_5_6 |
  1   | -157.3051381643 | -157.3051381643 |         G_3_4_6_7 |
  1   | -157.3057145289 | -157.3057145289 |         G_3_5_6_7 |
  1   | -157.3074706011 | -157.3074706011 |         G_3_6_7_8 |
  1   | -157.3078019201 | -157.3078019201 |         G_6_7_8_9 |
 -2   | -118.2701325572 |  236.5402651145 |           G_1_2_3 |
 -2   | -118.2684223683 |  236.5368447365 |           G_2_3_4 |
 -2   | -118.2687952758 |  236.5375905517 |           G_2_3_5 |
 -3   | -118.2675117406 |  354.8025352217 |           G_2_3_6 |
 -1   | -118.2706308152 |  118.2706308152 |           G_3_4_5 |
 -2   | -118.2691546284 |  236.5383092567 |           G_3_4_6 |
 -2   | -118.2695519038 |  236.5391038076 |           G_3_5_6 |
 -3   | -118.2703538608 |  354.8110615825 |           G_3_6_7 |
 -1   | -118.2698076315 |  118.2698076315 |           G_6_7_8 |
  3   |  -79.2318269995 | -237.6954809985 |             G_2_3 |
  1   |  -79.2336613594 |  -79.2336613594 |             G_3_4 |
  1   |  -79.2340541054 |  -79.2340541054 |             G_3_5 |
  3   |  -79.2325730649 | -237.6977191947 |             G_3_6 |
 -1   |  -40.1984657167 |   40.1984657167 |               G_3 |
------------------------------------------------------------
```

```
Total bonded energy for SCF = -352.4867942522
-----------------------------------------------------------------------
```

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

```
Energy of Dimer|Energy of Mono1|Energy of Mono2|Nonbonded I.E.|Nonbonded I.E.|Dimer Names|
---------------|---------------|---------------|--------------|--------------|-----------|
   (hartree)   |     (hartree) |     (hartree) |    (hartree) |   (kcal/mol) |           |
---------------|---------------|---------------|--------------|--------------|-----------|
  -80.39565616 |  -40.19782045 |  -40.19788569 |   0.00004998 |   0.03136212 |   G_1_G_7 |
  -80.39556601 |  -40.19782045 |  -40.19773805 |  -0.00000752 |  -0.00471729 |   G_1_G_8 |
  -80.39573075 |  -40.19782045 |  -40.19790816 |  -0.00000214 |  -0.00134471 |   G_1_G_9 |
  -80.39444135 |  -40.19672185 |  -40.19773805 |   0.00001855 |   0.01163874 |   G_2_G_8 |
  -80.39463707 |  -40.19672185 |  -40.19790816 |  -0.00000706 |  -0.00443044 |   G_2_G_9 |
  -80.39635460 |  -40.19846572 |  -40.19790816 |   0.00001928 |   0.01209736 |   G_3_G_9 |
  -80.39571052 |  -40.19802493 |  -40.19773805 |   0.00005245 |   0.03291412 |   G_4_G_8 |
  -80.39594110 |  -40.19802493 |  -40.19790816 |  -0.00000801 |  -0.00502865 |   G_4_G_9 |
  -80.39568849 |  -40.19800026 |  -40.19773805 |   0.00004982 |   0.03126424 |   G_5_G_8 |
  -80.39591563 |  -40.19800026 |  -40.19790816 |  -0.00000721 |  -0.00452506 |   G_5_G_9 |


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      -0.3125600000
H         2.6997800000      -0.3550810000      -1.4162340000
H         3.4122590000      -1.2373920000      -0.0652510000
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

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

$set_lssmf
  level 3
  nb_level 1
  method_name [scf;mp2;ccd;ccsd;ccsd(t)]
  mode sow
$end_lssmf
```

The `lssmf.chk` file is printed out as below.

```
    Ngroup: 7
    Nnonbfrag: 6
    Nbfrag: 7

    Dimer    Mono1    Mono2
    ========================
    1         1         5
    2         1         6
    3         1         7
    4         2         6
    5         2         7
    6         3         7

    Coefficients  --->  Label of bonded frags
```

```
    ************          ********************
          1          --->      G_1_2_3_4
          1          --->      G_2_3_4_5
          1          --->      G_3_4_5_6
          1          --->      G_4_5_6_7
         -1          --->      G_2_3_4
         -1          --->      G_3_4_5
         -1          --->      G_4_5_6
```

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.

```
    ***********************************************************************
                            CCSD(T) ENERGY
    ***********************************************************************


    Coeff |          Energy |    Coeff*Energy | Bonded Group Name |
    ------|-----------------|-----------------|-------------------|
        1 | -157.9784894936 | -157.9784894936 |         G_1_2_3_4 |
        1 | -157.9782837483 | -157.9782837483 |         G_2_3_4_5 |
        1 | -157.9787063574 | -157.9787063574 |         G_3_4_5_6 |
        1 | -157.9796304517 | -157.9796304517 |         G_4_5_6_7 |
       -1 | -118.7796321247 |  118.7796321247 |           G_2_3_4 |
       -1 | -118.7800829289 |  118.7800829289 |           G_3_4_5 |
       -1 | -118.7800993839 |  118.7800993839 |           G_4_5_6 |
    ----------------------------------------------------------------
     Total bonded energy for CCSD(T) = -275.5752956135
    ----------------------------------------------------------------


Energy of Dimer|Energy of Mono1|Energy of Mono2|Nonbonded I.E.|Nonbonded I.E.|Dimer Names|
---------------|---------------|---------------|--------------|--------------|-----------|
   (hartree)   |     (hartree) |     (hartree) |    (hartree) |   (kcal/mol) |           |
---------------|---------------|---------------|--------------|--------------|-----------|
  -80.77367284 |  -40.38692204 |  -40.38640503 |  -0.00034577 |  -0.21697453 |   G_1_G_5 |
  -80.77342760 |  -40.38692204 |  -40.38642795 |  -0.00007761 |  -0.04870338 |   G_1_G_6 |
  -80.77387251 |  -40.38692204 |  -40.38692804 |  -0.00002244 |  -0.01407833 |   G_1_G_7 |
  -80.77296019 |  -40.38634832 |  -40.38642795 |  -0.00018392 |  -0.11541067 |   G_2_G_6 |
  -80.77331843 |  -40.38634832 |  -40.38692804 |  -0.00004207 |  -0.02639750 |   G_2_G_7 |
  -80.77343675 |  -40.38632403 |  -40.38692804 |  -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`İn 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.

```
$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     -0.0652510000
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

$set_globals
  jobtype energy
  mol_frag true
$end_globals

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

$set_frag_external
```

```
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
}
$end_frag_external
```

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
```

```
}
EG_1_2_3_4=energy('CCSD(T)', molecule=G_1_2_3_4)
E1=get_variable('SCF TOTAL ENERGY')
print_out('==>    SCF Energy of Group G_1_2_3_4 =  %20.12f   \n' % E1)
E2=get_variable('MP2 TOTAL ENERGY')
print_out('==>    MP2 Energy of Group G_1_2_3_4 =  %20.12f   \n' % E2)
E3=get_variable('CCD TOTAL ENERGY')
print_out('==>    CCD Energy of Group G_1_2_3_4 =  %20.12f   \n' % E3)
E4=get_variable('CCSD TOTAL ENERGY')
print_out('==>    CCSD Energy of Group G_1_2_3_4 =  %20.12f   \n' % E4)
E5=get_variable('CCSD(T) TOTAL ENERGY')
print_out('==>    CCSD(T) Energy of Group G_1_2_3_4 =  %20.12f   \n' % E5)
clean()
```

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         3.2171710000      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     -0.0652510000
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

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

$set_lssmf
  level 3
  nb_level 3
  method_name [scf;mp2]
  mode sow
$end_lssmf
```

The `lssmf.chk` file is printed out as below.

```
    Ngroup: 7
    Nnonbfrag: 5
    Nbfrag: 7

          Dimer            Coefficients
      **************      **************
        1 <-> 5_6_7              1
        7 <-> 1_2_3              1
      1_2 <-> 6_7                1
        1 <-> 6_7               -1
        7 <-> 1_2               -1


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


    Coefficients  --->  Label of bonded frags
    ************        ********************
```

```
          1          --->      G_1_2_3_4
          1          --->      G_2_3_4_5
          1          --->      G_3_4_5_6
          1          --->      G_4_5_6_7
         -1          --->      G_2_3_4
         -1          --->      G_3_4_5
         -1          --->      G_4_5_6
```

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.

```
    ********************************************************************
                            MP2 ENERGY
    ********************************************************************


    Coeff |          Energy |    Coeff*Energy | Bonded Group Name |
    ------|-----------------|-----------------|-------------------|
        1 | -157.8967782614 | -157.8967782614 |         G_1_2_3_4 |
        1 | -157.8965730873 | -157.8965730873 |         G_2_3_4_5 |
        1 | -157.8970383126 | -157.8970383126 |         G_3_4_5_6 |
        1 | -157.8978930447 | -157.8978930447 |         G_4_5_6_7 |
       -1 | -118.7158761433 |  118.7158761433 |           G_2_3_4 |
       -1 | -118.7163633621 |  118.7163633621 |           G_3_4_5 |
       -1 | -118.7164024508 |  118.7164024508 |           G_4_5_6 |
    ------------------------------------------------------------
     Total bonded energy for MP2 = -275.4396407497
    ------------------------------------------------------------


Energy of Dimer|Energy of Mono1|Energy of Mono2|Nonbonded I.E.|Coeff|I.E.*Coeff|Dimer Names|
---------------|---------------|---------------|--------------|-----|----------|-----------|
     (hartree) |     (hartree) |     (hartree) |    (hartree) |     |(kcal/mol)|           |
---------------|---------------|---------------|--------------|-----|----------|-----------|
 -159.07701523 |   -40.35941955 | -118.71718008 |   -0.00041560 |   1 |     -0.26|G_1_G_5_6_7|
 -159.07620847 |   -40.35942736 | -118.71655498 |   -0.00022613 |   1 |     -0.14|G_7_G_1_2_3|
 -159.07335335 |   -79.53647925 |   -79.53659144 |   -0.00028266 |   1 |     -0.18|G_1_2_G_6_7|
 -119.89610130 |   -40.35941955 |   -79.53659144 |   -0.00009031 |  -1 |      0.06|  G_1_G_6_7|
 -119.89595802 |   -40.35942736 |   -79.53647925 |   -0.00005141 |  -1 |      0.03|  G_7_G_1_2|
-------------------------------------------------------------------------------------------
  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
```

```
0 1
O      -9.068110    0.722660   -1.519780 -1
O      -7.011900   -0.029570   -1.195380
C      -8.320340   -0.239900   -1.439050
C      -8.824910   -1.662870   -1.642330
C     -10.350260   -1.707510   -2.035820
C     -10.872490   -3.096450   -2.509370
C     -11.305090   -1.091120   -1.024200
H     -10.401440   -1.065870   -2.945110
O     -10.882560   -0.675020    0.185400
O     -12.482580   -0.951530   -1.316430 -1
N     -11.205560   -4.042800   -1.436050 +1
H     -11.613770   -4.901990   -1.870310
H     -11.924410   -3.639390   -0.794140
H     -10.150260   -3.553080   -3.221100
H     -11.811830   -2.916030   -3.080360
N      -8.472060   -2.490180   -0.477890 +1
H      -8.251990   -2.050510   -2.515240
H      -9.079230   -2.293020    0.347010
H      -7.477340   -2.320770   -0.202850
H     -10.379930   -4.342940   -0.885470
H      -8.505930   -3.498690   -0.730410
$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
```

```
@@@

$SET_MOLECULE G4
0 1
C        -10.3502600000         -1.7075100000         -2.0358200000
H        -10.4014400000         -1.0658700000         -2.9451100000
H         -9.2764938816         -1.6760857895         -1.7588237500  #CAP
H        -10.7178824342         -2.6852506579         -2.3691742763  #CAP
H        -11.0224100658         -1.2736038816         -1.3236927632  #CAP
$END_MOLECULE

@@@

$SET_MOLECULE G5
0 1
C        -10.8724900000         -3.0964500000         -2.5093700000
H        -10.1502600000         -3.5530800000         -3.2211000000
H        -11.8118300000         -2.9160300000         -3.0803600000
H        -10.5048675658         -2.1187093421         -2.1760157237  #CAP
H        -11.1149287075         -3.7852897959         -1.7281098639  #CAP
$END_MOLECULE

@@@

$SET_MOLECULE G6
-1 1
C        -11.3050900000         -1.0911200000         -1.0242000000
O        -10.8825600000         -0.6750200000          0.1854000000
O        -12.4825800000         -0.9515300000         -1.3164300000
H        -10.6329399342         -1.5250261184         -1.7363272368  #CAP
$END_MOLECULE

@@@

$SET_MOLECULE G7
1 1
N        -11.2055600000         -4.0428000000         -1.4360500000
H        -11.6137700000         -4.9019900000         -1.8703100000
H        -11.9244100000         -3.6393900000         -0.7941400000
H        -10.3799300000         -4.3429400000         -0.8854700000
H        -10.9744502041         -3.3861489796         -2.1808026531  #CAP
$END_MOLECULE

@@@
```

14.3 FRAGMENT Module Options

- **MODE**
  Type of the mode.
  TYPE: Sring
  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: Sring
  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: n 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: n 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: n 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: n 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: n User-defined value.

References

[1] J. L. Whitten, J. Chem. Phys. **58**, 4496 (1973).

[2] B. I. Dunlap, J. W. D. Connolly, and J. R. Sabin, J. Chem. Phys. **71**, 3396 (1979).

[3] M. Feyereisen, G. Fitzgerald, and A. Komornicki, Chem. Phys. Lett. **208**, 359 (1993).

[4] O. Vahtras, J. Almlöf, and M. W. Feyereisen, Chem. Phys. Lett. **213**, 514 (1993).

[5] A. P. Rendell and T. J. Lee, J. Chem. Phys. **101**, 400 (1994).

[6] F. Weigend, Phys. Chem. Chem. Phys. **4**, 4285 (2002).

[7] A. Sodt, J. E. Subotnik, and M. Head-Gordon, J. Chem. Phys. **125**, 194109 (2006).

[8] H.-J. Werner and M. Schütz, J. Chem. Phys. **135**, 144116 (2011).

[9] A. E. DePrince and C. D. Sherrill, J. Chem. Theory Comput. **9**, 2687 (2013).

[10] U. Bozkaya, J. Chem. Theory Comput. **10**, 2371 (2014).

[11] U. Bozkaya, J. Chem. Phys. **141**, 124108 (2014).

[12] U. Bozkaya, J. Chem. Theory Comput. **10**, 4389 (2014).

[13] U. Bozkaya, J. Chem. Theory Comput. **12**, 1179 (2016).

[14] U. Bozkaya, Phys. Chem. Chem. Phys. **18**, 11362 (2016).

[15] U. Bozkaya, J. Chem. Phys. **144**, 144108 (2016).

[16] U. Bozkaya and C. D. Sherrill, J. Chem. Phys. **144**, 174103 (2016).

[17] U. Bozkaya and C. D. Sherrill, J. Chem. Phys. **147**, 044104 (2017).

[18] U. Bozkaya, J. Comput. Chem. **39**, 351 (2018).

[19] U. Bozkaya, J. Chem. Theory Comput. **15**, 4415 (2019).

[20] U. Bozkaya, Int. J. Quant. Chem. **121**, e26623 (2021).

[21] T. H. Dunning, J. Chem. Phys. **90**, 1007 (1989).

[22] D. E. Woon and T. H. Dunning, J. Chem. Phys. **103**, 4572 (1995).

[23] T. Helgaker, P. Jørgensen, and J. Olsen, *Molecular Electronic Structure Theory*, 1 ed. (John Wiley & Sons, New York, 2000), pp. 336–426.

[24] H. B. Schlegel and M. J. Frisch, Int. J. Quant. Chem. **54**, 83 (1995).

[25] F. Weigend, A. Köhn, and C. Hättig, J. Chem. Phys. **116**, 3175 (2002).

[26] B. P. Pritchard, D. Altarawy, B. Didier, T. D. Gibson, and T. L. Windus, J. Chem. Inf. Model. **59**, 4814 (2019).

[27] G. Chaban, M. W. Schmidt, and M. S. Gordon, Theor. Chem. Acc. **97**, 88 (1997).

[28] M. Wolfsberg and L. Helmholtz, J. Chem. Phys. **20**, 837 (1952).

[29] M. Guest and V. R. Saunders, Mol. Phys. **28**, 819 (1974).

[30] C. C. J. Roothaan, Rev. Mod. Phys. **32**, 179 (1960).

[31] E. R. Davidson, Chem. Phys. Lett. **21**, 565 (1973).

[32] J. Binkley, J. Pople, and P. Dobosh, Mol. Phys. **28**, 1423 (1974).

[33] R. McWeeny and G. Diercksen, J. Chem. Phys. **49**, 4852 (1968).

[34] K. Fægri and R. Manne, Mol. Phys. **31**, 1037 (1976).

[35] B. N. Plakhutin, E. V. Gorelik, and N. N. Breslavskaya, J. Chem. Phys. **125**, 204110 (2006).

[36] G. M. J. Barca, C. Bertoni, L. Carrington, D. Datta, N. D. Silva, J. E. Deustua, D. G. Fedorov, J. R. Gour, A. O. Gunina, E. Guidez, T. Harville, S. Irle, J. Ivanic, K. Kowalski, S. S. Leang, H. Li, W. Li, J. J. Lutz, I. Magoulas, J. Mato, V. Mironov, H. Nakata, B. Q. Pham, P. Piecuch, D. Poole, S. R. Pruitt, A. P. Rendell, L. B. Roskop, K. Ruedenberg, T. Sattasathuchana, M. W. Schmidt, J. Shen, L. Slipchenko, M. Sosonkina, V. Sundriyal, A. Tiwari, J. L. G. Vallejo, B. Westheimer, M. Włoch, P. Xu, F. Zahariev, and M. S. Gordon, J. Chem. Phys. **152**, 154102 (2020).

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

[38] F. Jensen, *Introduction to computational chemistry* (John wiley & sons, Chichester, 2017), pp. 316–338.

[39] W. H. Miller, N. C. Handy, and J. E. Adams, J. Chem. Phys. **72**, 99 (1980).

[40] B. Ermis, A. Unal, E. Soydas, and U. Bozkaya (unpublished).

[41] D. G. A. Smith, L. A. Burns, A. C. Simmonett, R. M. Parrish, M. C. Schieber, R. Galvelis, P. Kraus, H. Kruse, R. D. Remigio, A. Alenaizan, A. M. James, S. Lehtola, J. P. Misiewicz, M. Scheurer, R. A. Shaw, J. B. Schriber, Y. Xie, Z. L. Glick, D. A. Sirianni, J. S. O'Brien, J. M. Waldrop, A. Kumar, E. G. Hohenstein, B. P. Pritchard, B. R. Brooks, H. F. Schaefer, A. Y. Sokolov, K. Patkowski, A. E. DePrince, U. Bozkaya, R. A. King, F. A. Evangelista, J. M. Turney, T. D. Crawford, and C. D. Sherrill, J. Chem. Phys. **152**, 184108 (2020).

[42] Y. Shao *et al.*, Molecular Physics **113**, 184 (2014).

[43] D. A. Matthews, L. Cheng, M. E. Harding, F. Lipparini, S. Stopkowicz, T.-C. Jagau, P. G. Szalay, J. Gauss, and J. F. Stanton, J. Chem. Phys. **152**, 214108 (2020).

[44] H.-J. Werner, P. J. Knowles, F. R. Manby, J. A. Black, K. Doll, A. Heßelmann, D. Kats, A. Köhn, T. Korona, D. A. Kreplin, Q. Ma, T. F. Miller, A. Mitrushchenkov, K. A. Peterson, I. Polyak, G. Rauhut, and M. Sibaev, J. Chem. Phys. **152**, 144107 (2020).

[45] F. Neese, WIREs Comput. Mol. Sci. **2**, 73 (2012).

[46] D. A. Clabo, W. D. Allen, R. B. Remington, Y. Yamaguchi, and H. F. Schaefer, Chem. Phys. **123**, 187 (1988).

[47] J. M. L. Martin, T. J. Lee, P. R. Taylor, and J.-P. François, J. Chem. Phys. **103**, 2589 (1995).

[48] U. Bozkaya, J. M. Turney, Y. Yamaguchi, H. F. Schaefer, and C. D. Sherrill, J. Chem. Phys. **135**, 104103 (2011).

[49] U. Bozkaya and C. D. Sherrill, J. Chem. Phys. **138**, 184103 (2013).

[50] U. Bozkaya, J. Chem. Phys. **135**, 224103 (2011).

[51] E. Soydaş and U. Bozkaya, J. Chem. Theory Comput. **9**, 1452 (2013).

[52] U. Bozkaya, J. Chem. Phys. **139**, 104116 (2013).

[53] U. Bozkaya and C. D. Sherrill, J. Chem. Phys. **141**, 204105 (2014).

[54] E. Soydaş and U. Bozkaya, J. Chem. Theory Comput. **11**, 1564 (2015).

[55] U. Bozkaya and C. D. Sherrill, J. Chem. Phys. **139**, 054104 (2013).

[56] E. Soydaş and U. Bozkaya, J. Comput. Chem. **35**, 1073 (2014).

[57] U. Bozkaya, A. Ünal, and Y. Alagöz, J. Chem. Phys. **153**, 244115 (2020).

[58] U. Bozkaya and H. F. Schaefer, J. Chem. Phys. **136**, 204114 (2012).

[59] E. Epifanovsky, D. Zuev, X. Feng, K. Khistyaev, Y. Shao, and A. I. Krylov, J. Chem. Phys. **139**, 134105 (2013).

[60] A. Ünal, T. Bozkaya, and U. Bozkaya, (unpublished) **?**, ? (2021).

[61] Y. Shao, L. Fusti-Molnar, Y. Jung, J. Kussmann, C. Ochsenfeld, S. T. Brown, A. T. B. Gilbert, L. V. Slipchenko, S. V. Levchenko, D. P. O'Neill, R. A. DiStasio, R. C. Lochan, T. Wang, G. J. O. Beran, N. A. Besley, J. M. Herbert, C. Y. Lin, T. V. Voorhis, S. H. Chien, A. Sodt, R. P. Steele, V. A. Rassolov, P. E. Maslen, P. P. Korambath, R. D. Adamson, B. Austin, J. Baker, E. F. C. Byrd, H. Dachsel, R. J. Doerksen, A. Dreuw, B. D. Dunietz, A. D. Dutoi, T. R. Furlani, S. R. Gwaltney, A. Heyden, S. Hirata, C. P. Hsu, G. Kedziora, R. Z. Khaliullin, P. Klunzinger, A. M. Lee, M. S. Lee, W. Liang, I. Lotan, N. Nair, B. Peters, E. I. Proynov, P. A. Pieniazek, Y. M. Rhee, J. Ritchie, E. Rosta, C. D. Sherrill, A. C. Simmonett, J. E. Subotnik, H. L. Woodcock, W. Zhang, A. T. Bell, A. K. Chakraborty, D. M. Chipman, F. J. Keil, A. Warshel, W. J. Hehre, H. F. Schaefer, J. Kong, A. I. Krylov, P. M. W. Gill, and M. Head-Gordon, Phys. Chem. Chem. Phys. **8**, 3172 (2006).