MOLCAS version 7.4

User's Manual

Contents

List of Figures

List of Tables

Section 1

Introduction to *MOLCAS*

1.1 *MOLCAS*, a quantum chemistry software

MOLCAS is a quantum chemistry software developed by scientists to be used by scientists. It is not primarily a commercial product and it is not sold in order to produce a fortune for its owner (the Lund University). The authors have tried in *MOLCAS* to assemble their collected experience and knowledge in computational quantum chemistry. *MOLCAS* is a research product and it is used as a platform by the scientists in the *MOLCAS* network in their efforts to develop new and improved computational tools in quantum chemistry. Several of the codes in the software have newly developed features and the user should not be surprised if a bug is found now and then.

The basic philosophy behind *MOLCAS* is to develop methods that will allow an accurate *ab initio* treatment of very general electronic structure problems for molecular systems in both ground and excited states. This is not an easy task. Our knowledge about how to obtain accurate properties for single reference dominated ground states is today well developed and *MOLCAS* contains a number of codes that can perform such calculations (MP2, CC, CPF, DFT etc). All these methods treat the electron correlation starting from a single determinant (closed or open shell) reference state. Such codes are today standard in most quantum chemistry program systems.

However, the basic philosophy of *MOLCAS* is to be able to treat, at the same level of accuracy also, highly degenerate states, such as those occurring in excited states, at the transition state in some chemical reactions, in diradicaloid systems, heavy metal systems, etc. This is a more difficult problem since the single determinant approach will not work well in such cases. The key feature of *MOLCAS* is the multiconfigurational approach. *MOLCAS* contains codes for general and effective multiconfigurational SCF calculations at the Complete Active Space (CASSCF) level, but also employing more restricted MCSCF wave functions (RASSCF). It is also possible, at this level of theory, to optimize geometries for equilibrium and transition states using gradient techniques and to compute force fields and vibrational energies.

However, even if the RASSCF approach is known to give reasonable structures for degenerate systems — both in ground and excited states — it is not in general capable of recovering more than a small fraction of the correlation energy. It is therefore necessary to supplement the multiconfigurational SCF treatment with a calculation of the dynamic correlation effects. In the earliest version of *MOLCAS*, this was achieved by means of the multireference (MR)

CI method. This method has, however, severe limitations in the number of electrons that can be correlated and the size of the reference space. It is not a method that can be used to study excited states of anything but small molecules. But here it has the capacity to produce very accurate wave functions and potential surfaces. The MRCI code of MOLCAS is used by many groups for this purpose. Today it is also possible to run the COLUMBUS MRCI code together with MOLCAS.

In the years 1986-90, a new method was developed, which can be used to compute dynamic electron correlation effects for multiconfigurational wave functions. It is based on second order perturbation theory and has been given the acronym CASPT2. It was included into the second version of MOLCAS. From the beginning it was not clear whether the CASPT2 method would be accurate enough to be useful in practice. However, as it turned out it was surprisingly accurate in a number of different types of applications. The CASPT2 approach has become especially important in studies of excited states and spectroscopic properties of large molecules, where no other *ab initio* method has, so far, been applicable. The method is based on second order perturbation theory and has therefore limitations in accuracy, but the error limits have been investigated in a large number of applications. The errors in relative energies are in almost all cases small and the results can be used for conclusive predictions about molecular properties in ground and excited states. Important application areas for the CASPT2 method are potential energy surfaces for chemical reactions, photochemistry, transition metal chemistry and heavy element chemistry. The method is under constant development. A multistate version is available, which allows the simultaneous study of several electronic states, including their interaction in second order. This code is especially useful in cases where two, or more energy surfaces are close in energy. We have for a number of years also tried to develop an analytical CASPT2 gradient code. For different reasons, this work is as yet unfinished. Instead we have in the present version (7.4) included a numerical procedure, which allows automatic geometry optimization at the CASPT2 level of theory. It is applicable to all states and systems for which the CASPT2 energy can be computed and can also be used to compute vibrational frequencies.

MOLCAS contains apart from the pure wave function codes, also the possibility to compute molecular properties, either as expectation values, or using finite perturbation theory. It is also possible to model solvent effects by adding a reaction field Hamiltonian (PCM). 7.4 also includes a new QM/MM model.

The program RASSI has the capacity to compute the interaction between several RASSCF wave functions based on different orbitals, which are in general non-orthonormal (nonorthogonal CI). RASSI is routinely used to compute transition dipole moments in spectroscopy, but can also be used, for example, to study electron transfer or other properties where it might be of value to use localized wave functions.

MOLCAS-5 added an important extension of the RASSI code. It now became possible to use it to compute spin-orbit interaction between different electronic states. Together with the Douglas-Kroll treatment of scalar relativistic effects, this new option opened up the entire periodic system for calculations at the CASSCF/CASPT2 level with MOLCAS. Preliminary studies for actinides and other heavy atom systems have been very promising [1]. This is today an important tool in a number of applications involving heavy atoms. The ANO basis set library has been extended to cover the entire periodic systems. The new ANOs (ANO-RCC) are produced with the inclusion of scalar relativistic effects and include correlation of semi-core orbitals [2, 3, 4, 5, 6].

MOLCAS-6 added a number of new features. It now became possible to perform DFT calculations, both for closed and open shell systems. Gradients became available for geometry optimizations, transition state searches, etc. The new software also included a module for computing local properties (multipole moments and polarizabilities), which are used to construct the NEMO force fields for MC/MD simulations of macromolecular systems, liquids, etc. With version 7.4 the NEMO module was taken out of the distribution version, but can be obtained by contacting the *MOLCAS* team.

The size of the systems that can be treated with *MOLCAS* have been limited due to limitations in storing two-electron integrals for large basis set. This limit has now been moved substantially to larger systems by the introduction of a Cholesky decomposition of the two-electron integrals. This feature is introduced in *MOLCAS*-7 at all levels of theory [7, 8, 9]. It speeds up all calculations by orders of magnitude and extends the size of the basis sets that can be used. The accuracy can be controled by the threshold used in the decomposition. The same approach can be used to generate RI auxiliary basis sets on the fly, which can then be used, for example to compute energy derivatives at the SCF, DFT, and RASSCF levels of theory.

It should finally be clearly stated that *MOLCAS* is not a black box tool. The user should be a chemist, with some knowledge about the different quantum chemical models in use today, their application areas and their inherent accuracy. He should also have a critical mind and not take a printed output for granted without checking that the results are consistent with the model he has employed. The skill to use *MOLCAS* effectively does not come immediately, but we have tried to help the user by providing together with this manual a book of examples, which explains how some different key projects were solved using *MOLCAS*. We are sure that the users will find them helpful in their own attempts to master the software and use it in the chemical applications. The MOLCAS group arranges regular MOLCAS workshops, which teaches how to use the software.

1.2 The *MOLCAS* Manuals

1.2.1 The three parts of the manual

The *MOLCAS* manual is divided in three different parts. Beginners are recommended to primarily consult the first part, the “Tutorials and Examples” part to get aquainted with *MOLCAS*. The Tutorials is a detailed and first-principles guide for the *MOLCAS* programs with easy explanations of the main features of the input keywords and output information. The Examples are a selection of more elaborated calculations performed with *MOLCAS*, where difficult situations are described and more detailed explanations are focused on the quantum chemical consequences of the use of the codes.

The second part, the “User’s Guide”, is a reference manual of the *MOLCAS* package. It contains the basic description of each of the *MOLCAS* molcas programs, the dependencies among the different modules, input keywords, and input and output files. It may be rather difficult to read for an unexperienced *MOLCAS* user.

Finally, anybody who is going to install and/or maintain *MOLCAS* needs to consult the third and last part, the “Installation Guide”.

An up to date version of the manual is available on the net in HTML and PostScript formats (`http://www.molcas.org`).

1.2.2 Notation

For clarity, some words are printed using special typefaces.

- Keywords, i.e. words used in input files, are typeset in the small-caps typeface, for example ENDOFINPUT.
- Programs (or modules) are typeset in the teletype typeface. This will eliminate some potential confusion. For example, when discussing the RASSCF method, regular uppercase letters are used, while the program will look like `RASSCF`.
- Files are typeset in the slanted teletype typeface, like *`InpOrb`*.
- Commands, unix or other, are typeset in a sans serif typeface, like ln -fs.
- Complete examples, like input files, shell scripts, etc, are typeset in the teletype typeface. There are two variants, one for input files and one for other examples. For input files the space (or blank) character is represented with the character ␣. This makes it easy for the reader to see the spaces, which are sometimes important characters. In all other cases the space character is handled in the normal fashion.

1.2.3 Molcas documentation

The following documentation is available for *MOLCAS* in a single book.

- *MOLCAS* version 7.4 — User's Guide.
- *MOLCAS* version 7.4 — Tutorials and Examples.
- *MOLCAS* version 7.4 — Installation Guide.

An additional documentation about the structure of *MOLCAS* package is available in a separate book "Molcas Programming Guide".

1.3 *MOLCAS*-7, new features and updates

Below is presented a list of the major new features of *MOLCAS* . These features comprise a number of new codes and introduction of new methods, but also considerable updates of many of the programs in *MOLCAS*. We keep some history, so that people who are using older versions of *MOLCAS* can get a feeling for what has happened on later versions

New features in 7.4 **LIST IS MISSING!!!**

New features in 7.2

- New codes and major updates:
  - pre-release version of GUI for input generation and molcas job submition (MING).

  - Module Seward has been split into Gateway (set up of molecular system) and Seward itself (computation of integrals).
  - Major improvements in runtime settings for the package, and new flags for molcas command
  - New manual for novice molcas users

- Performance enhancements:
  - A new version of GA has been included.
  - Default integral thresholds are now changed to 1.0D-10.
  - RI code has been improved

- New features in existing codes:
  - The exchange-hole dipole moments in LoProp code
  - Better handling of sypersymmetry in RASSCF code
  - Localized natural orbitals in Localisation code
  - BSSE calculations in SCF code
  - A second finite nuclei charge distribution model, the so-called modified Gaussian charge distribution, has been implemented
  - Frequency calculations for MBPT2
  - The ESPF module can be used in order to compute electrostatic potential derived charges
  - Frozen Natural Orbital approach in CASPT2
  - On-the-fly generation of RI auxilary basis set
  - Flexible selection of orbitals in GRID_IT
  - New features in gv code: visualization of molden files, selection of atomic groups, symmetry operations

- Changes in usage of the package:
  - No shell scripts are needed to run molcas.
  - New EMIL commands for file handling
  - Control of the print level of the code

- Installation and tools
  - New tools for memory and I/O profiling
  - New configuration files has been included

New features in 7.0

- New codes and major updates:
  - CHOLESKY - a new approach to ab initio and first principle QM methods free from explicit two-electron integrals. SCF/DFT, RASSCF, RASSI and MP2 energy calculation can now be done with considerable improvement of performance and with controlled accuracy of the results.

  - The 1-center approximation of the Cholesky decomposition, 1-CCD
  - Resolution of Identity (RI)/ Density fitting (DF) scheme for SCF, DFT, CASSCF, RASSI and CASPT2
  - The CASPT2 module can be used in connection with Cholesky and RI/DF approximations, allowing for the treatment of larger systems
  - Update of guessorb code
  - Electrostatic potential fitted (ESPF) QM/MM interface for SCF, DFT, CASSCF, CASPT2, and CC. ESPF analytic gradients for SCF, DFT, and CASSCF.
  - Gradients for 'pure' DFT for the 1-CCD, and RI/DF approximations
  - Scaled Opposite-Spin (SOS) and Scaled Spin Component (SCS) MP2 are implemented when using Cholesky or RI/DF approximation.
  - NEMO program: fitting of potential surfaces, energy optimizations, potential curves and simulation parameters.
  - interface to MOLSIM code
  - Major update for GUI code gv, with a possibility to edit coordinates and visually select active space for RASSCF calculations.
  - A new program, EXPBAS, has been introduced that allows expanding an orbital file from a small to a larger basis set.
  - Several different procedures for constructing localized orbitals have been implemented. Among them is one based on a Cholesky decomposition of the density matrix.

- Performance enhancements:
  - Use of external blas libraries: lapack, GotoBLAS, Atlas, Intel MKL, ACML
  - New version of GA has been included.
  - Improved diagonalization routines and improved convergence in scf and rasscf
  - Some size limits in RASSCF and CASPT2 have been increased or eliminated.
  - Automatic generation of starting orbitals for arbitrary valence and ECP basis sets.

- New features in existing codes:
  - Natural orbitals for UHF calculations. Can, for example be used as starting orbitals for RASSCF.
  - Natural Bond Order (NBO) based on the LoProp partitioning.
  - Arbitrary order Douglas-Kroll-Hess (DKH) transformation to include scalar relativistic effects.
  - Picture-change-corrected electric potential, electric field, and electric field gradient properties.
  - Automatic generation of rydberg orbitals in genano.
  - RASSI can compute g-tensors.
  - CASPT2 is able to run with Cholesky vectors instead of integrals.
  - Transverse constraint for geometry optimizations.

  - Numerical gradients for several methods.
  - Numerical IR intensities for Numerical Hessian.
  - Computation of charge capacitances for bonds using Loprop.
  - Localized exchange-hole dipole moments in Loprop.
  - Possibility to use loprop with user-defined densities.
  - Evaluation of transition density between two states.
  - Mulliken type multicenter multipole expansion and localized polarizablilites based on the uncoupled HF approach.
  - Several improvements and enhancements in the visualization program GV.
  - The ANO-RCC basis set is now complete covering all atoms H-Cm.
  - The GUESSORB facility is now included in SEWARD, which automatically produces starting orbitals for arbitrary basis sets.

- Changes in usage of the package:
  - Improvements in MOLCAS input language.
  - Molcas job can be submitted without shell scripts.
  - The programs are making extensive use of the runfile to simplify the input and eliminate unnecessary inputs.
  - automatic saving of output files (molden files, and orbital files)
  - The starting orbitals for RASSCF can be taken from a number of sources (Guessorb, runfile, etc.), and this is done in a semi-intelligent way unless specified in user input.
  - simplified RASSCF input: number of orbitals, spin, etc can sometimes be deduced by the program from information available on the runfile or an orbital file. One can use CHARGE instead of the number of active electrons.
  - If used in multiple runs in one job, the RASSCF automatically selects suitable individual names for the JOBIPH files. The choice can be overridden by keyword input, but if not, it matches the default selection of JOBIPH names in RASSI.
  - RASSI can use default selection of JOBIPH names, when used together with multiple RASSCF runs in one job.
  - RASSCF can use natural orbitals from a preceeding UHF calculation as input orbitals.

- Installation and tools
  - improved installation procedure, with possibility to select compilers, BLAS libraries, and parallel environment.
  - Configuration files for new compilers, including gfortran, g95, SunStudio
  - Configuration files for OpenMP parallelization.
  - Tools for extracting information from RUNFILE and JOBIPH files.

1.4 Acknowledgment

The *MOLCAS* project is carried out by the Lund University quantum chemistry group supported by the Swedish Science Research Council (VR).

The MOLCAS group is a member of the Linneaus project *Organising Molecular Matter, OMM.*

The MOLCAS group acknowledges the contributions from the scientists in the MOLCAS Network, who are memebrs of the MOLCAS developers team.

The Lund University Center for Scientific and Technical Computing (LUNARC) has provided computer resources for the project.

1.5 Citation for *MOLCAS*

The recommended citation for *MOLCAS* Version 7.4 is:

G. Karlström, R. Lindh, P.-Å. Malmqvist, B. O. Roos, U. Ryde, V. Veryazov, P.-O. Widmark, M. Cossi, B. Schimmelpfennig, P. Neogrady, L. Seijo, *Computational Material Science*, **28**, 222 (2003) [10].

The following persons have contributed to the development of the *MOLCAS* software:

K. Andersson, F. Aquilante, A. Bernhardsson, M. R .A. Blomberg, D. L. Cooper, M. Cossi, A. Devarajan, L. De Vico, N. Ferré, M. P. Fülscher, A. Gaenko, L. Gagliardi, G. Ghigo, C. de Graaf, B. A. Hess, D. Hagberg, A. Holt, G. Karlström, J. W. Krogh, R. Lindh, P.-Å. Malmqvist, P. Neogrády, J. Olsen, T. B. Pedersen, J. Raab, M. Reiher, B. O. Roos, U. Ryde, B. Schimmelpfennig, L. Seijo, L. Serrano-Andrés, P. E. M. Siegbahn, J. Stålring, T. Thorsteinsson, V. Veryazov, P.-O. Widmark, and A. Wolf.

1.6 Web addresses

We can be contacted on the web:
URL: `http://www.molcas.org`

1.7 Disclaimer

MOLCAS is shipped on an "as is" basis without warranties of any kind. The authors of *MOLCAS*-7 therefore assume no responsibility of any kind from the use of the supplied material. Permission is granted to use the package, but not to reproduce or redistribute any part of this material by any means. Material in the software may not be included in any commercial product. The authors reserve the right to change plans and existing material without notice.

Section 2

Introduction to the *MOLCAS* manual

This manual is designed for use with the *ab initio* quantum chemistry software package Molcas 7.4 [11] developed at the by the *MOLCAS* team all around the world, and that it has its base and origin in the Department of Theoretical Chemistry, Lund University, Sweden. *MOLCAS* is designed for use by theoretical chemists. It requires knowledge of the chemistry involved in the calculations in order to produce and interpret the results. The package is moderately difficult to use because of this 'knowledge requirement' but the results are often more meaningful than those produced by "blackbox" packages which are not as chemically precise in their input.

The *MOLCAS* manual is divided in several parts to facilitate its use. Initially, a Help Guide for novice and intermediate users is available, containing a series of tutorials and examples in which the use of the code is explained and the quantum chemical insights highlighted. Firstly, a short introductory guide: "How to run *MOLCAS* in a nutshell" is designed for all those who want an immediate start. Just the basics environment definitions and the most simple runs are collected, as well as the basic descriptions of how to collect the obtained information, from output and from the graphical tools. Secondly, the reader can find two types of introductory tutorials: problem-based and program specific tutorials. Problem-based tutorials are exercises focused to solve a simple quantum chemical project containing all the required input files, for instance, computing the electronic energy of a molecule at different levels of theory, optimizing a minimum or a transition state in the ground state of a chemical system, computing an excited state, etc. The inputs in this section can be found in the directory $MOLCAS/doc/samples/problem_based_tutorials and are those employed in the *MOLCAS* workshops that the *MOLCAS* team is organizing during the last years More complex cases can be found in the examples chapter. Another type of tutorials can be then found for most of the program modules available. These tutorials are designed for the first time user. Simple and easy to follow examples are presented for many of the modules contained in *MOLCAS*. The systems covered are not necessarily calculated with suitable methods or producing any significant results. There are, however, tips for the beginner and actual input file formats. Thirdly, there is a number of examples. These are outlines of actual research performed using *MOLCAS*. The approach to a research project is outlined including input files and shell scripts. More importantly, however, the value of the calculations is evaluated and advanced features of 7.4 are used and explained to improve the value of the results.

The 7.4 User's Guide contains a complete listing of the input keywords for each of the program modules and a information regarding files used in each calculation. Here the user will find all keywords that can be used together with a specific program and thus how to set up the input for a MOLCAS run.

The MOLCAS documentation finishes with an Installation Guide that describes simple and more complex aspects on how to install, tailor, and control the MOLCAS package.

Part I

Tutorials and Examples

Section 3

How to run *MOLCAS* in a nutshell

3.1 Introduction

Start running *MOLCAS* 7.4 just requires a small number of operations. This section of the manual, "How to run *MOLCAS* in a nutshell", is addressed to those who want to immediately run some simple calculation to become familiar with the program. Here we include the basic hints to begin with: setting the proper environment, building some simple inputs, running the calculation, and extracting the information obtained.

3.2 Environment Setup

The environment variable MOLCAS as well as the *MOLCAS* driver named ***molcas*** must be available to run *MOLCAS*. The variable MOLCAS points to the root directory of the *MOLCAS* installation. It can be defined by, for instance, the command

```
export MOLCAS=/home/molcas/molcas.version
```

The location of the *MOLCAS* driver is defined at installation time and is typically located in */usr/local/bin* ,*$HOME/bin*, etc. Check that such directory is included in your path. Otherwise you can extend your path (e.g. if molcas is in *$HOME/bin*:

```
export PATH=$PATH:$HOME/bin
```

It may be also convenient to define variables such as WorkDir, directory for intermediate files, or Project, name of the project:

```
export Project=MyMolecule
```

although *MOLCAS* will give them default values, as for the other environment variables (see section). All those definitions can be done directly or by creating a shell script.

3.3 Customization of molcas execution

MOLCAS has a flexible control for the organizing filenames and directories used during a calculation. The default values used for customization can be altered either by shell variables, or, preferably by a resource file *`molcasrc`*. A command molcas setuprc provides a guided help if user wants to create such file.

The terminology used in this chapter:

- `LOG` - output file and error file, produced by Molcas,
- `PN` - Project name, a name used for the file naming, e.g. `RUNFILE` used in the calculation will be named as `PN`.Runfile,
- `WDN` - WorkDir name, a name of directory used for temporary/binary files produced by molcas,
- `SDA` - scratch disk area, a path to a parent directory for `WDN`s. *`WorkDir`* variable used in the molcas manual is constructed as `SDA/WDN`,
- `SD` - submit directory, the directory where molcas command was issued. Note, that in this tutorial we will assume that the input file is located in `SD`,
- `OD` - output directory, the directory used for storage of extra output files, such as Orbital files, molden files.

It is quite important to understand, that if you have two consecutive runs of molcas, which are used the same scratch area (WorkDir), and the same project name, Molcas will try to reuse intermediate data, e.g. integrals and orbitals, in order to make a restart of your calculation. It might save a time, but from other hand, it can be dangerous if two calculations are not compatible.

Assuming that *`molcasrc`* is not exist, and no environment is set, the command molcas inputfile will use the following defaults: `LOG` is printing to the screen, `OD` is equal to `SD`, `PN` is constructed from the the name of *`inputfile`* by removing the suffix (before the last . (dot) character), `SDA` is equal to */tmp/*, `WDN` is constructed from `PN` by adding a random suffix.

For example if a user issued commands:

```
cd /home/joe/projects/water
vi H2O.DFT.input
molcas H2O.DFT.input
```

it will generate files:

```
/home/joe/projects/water/H2O.DFT.ScfOrb
/home/joe/projects/water/H2O.DFT.scf.molden
...
/tmp/H2O.DFT.15014/H2O.DFT.RunFile
...
```

If a flag -f is used in a Molcas command, `LOG` files will be stored in `SD` directory with a name `PN`.log and `PN`.err.

Project name can be set in a shell script running molcas, or included into molcas command: molcas Project=water H2O.DFT.input will change the default value for `PN` to water.

If `MOLCAS_WORKDIR` variable is set (a part of molcas command or included into *`molcasrc`* file), the name of WorkDir will NOT be random, but determined by the `PN`.

Example:

```
cd /home/joe/projects/water
vi H2O.DFT.input
molcas MOLCAS_WORKDIR=/tmp Project=water -f H2O.DFT.input
```

it will generate files:

```
/home/joe/projects/water/water.log
/home/joe/projects/water/water.ScfOrb
...
/tmp/water/water.RunFile
...
```

To get more options to control the behavior of molcas, user should run molcas setuprc script. File *`molcasrc`* can be used to set global preferences for *MOLCAS* package, or/and to set up user preferences (in the later case *`setuprc`* script creates a *`molcasrc`* file in a user *`HOME`* directory.

An example of molcasrc file for using *`/scratch`* area as a parent for WorkDirs, remove WorkDir before a calculation, and keep it when calculation finished, and use Project name generated for the the name of the input file:

```
# Version 1.0
MOLCASMEM=256
MOLCAS_WORKDIR=/scratch
MOLCAS_NEW_WORKDIR=YES
MOLCAS_KEEP_WORKDIR=YES
MOLCAS_PROJECT=NAME
```

Once the *`molcasrc`* is created, user usually don't need to use any shell scripts, or environment variables to run molcas.

3.4 *MOLCAS* Command-Line Help System

Just by typing molcas help you get access to *MOLCAS* Command-Line Help System. There are different options:

- molcas help produces a list of available programs and utilities.
- molcas help module yields the list of keywords of the program `MODULE`.
- molcas help module keyword offers the detailed description of the keyword.
- molcas help -t text displays a list of keywords that contain the text word in their description.

3.5 Input Structure and EMIL Commands

MOLCAS has a modular program structure. The easiest way to run calculations is to prepare an input file in which the different programs will be executed sequentially when the expression &module is found. After such a label the keywords of the program, if needed, follow, one on each line or separated by ;. Apart from the specific program input and keywords, *MOLCAS* incorporates certain commands (see section on EMIL commands) that allow operations such as to loop over the modules, restrict some of the executions, change variables or substitute certain unix commands.

3.6 Basic Examples

3.6.1 A simple calculation on water

Start by preparing a file containing the cartesian coordinates of the molecule, in this case water (this can be done by using any graphical interface program, for instance the *MOLCAS* gv module, as shown later):

```
␣3
Angstrom
␣O␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000
␣H␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284
␣H␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284
```

which we will name ***`water.xyz`***. In the same directory we prepare the input for the *MOLCAS* run. We can name it ***`water.input`***:

```
␣&gateway
␣coord=water.xyz
␣basis=sto-3g
␣&seward
␣&scf
```

The module `GATEWAY` compiles the geometric parameters of the molecule (added from the external file in this case) and the basis set definition. Module `SEWARD` computes the integrals and identifies, if any, the molecular symmetry, and `SCF` computes the Hartee-Fock wave function. For convenience just define:

export Project=water

and run the calculation, for instance:

molcas water.input -o water.out -e water.err

The file ***`water.out`*** contains now the output of the calculation and ***`water.err`*** the error messages, if any. In the same directory we can find other files such as ***`water.scf.molden`*** or ***`water.grid`*** (the last one only if you add at the end of the input the keyword GRID_IT) that will help to analyze the results graphically with the *MOLCAS* utility molcas gv (the freeware program `MOLDEN` can also be used). We will see their use below.

In the case of an open-shell case we can do a UHF (or UDFT) calculation with the `SCF` program or moving to the more powerful CASSCF approach. Below, we display two examples for UDFT, approximate doublet (setting the charge to +1), and a triplet state (using keyword ZSPIN to specify that there are two more α than β electrons) states, respectively, even if they are not pure spin functions:

```
␣&gateway
␣coord=water.xyz
␣basis=sto-3g
␣&seward
␣&scf
␣charge=+1
␣uhf;␣ksdft=b3lyp
```

```
␣&gateway
␣coord=water.xyz
␣basis=sto-3g
␣&seward
␣&scf
␣zspin=2
␣uhf;␣ksdft=b3lyp
```

3.6.2 Using a z-Matrix

Instead of cartesian coordinates we may want to define a z-matrix. In this example this is placed within the default input file (name it now ***ethanol.input***, with the definition of the basis set made independently for each of the atoms, using here the labeling defined by the program (see section):

```
&gateway
zMat
H.ano-s-mb
C.ano-s-mb
O.ano-s-mb
End␣of␣basis
C1
O2␣1␣1.3350
H3␣1␣1.0890␣2␣120.0
H4␣1␣1.0890␣2␣120.0␣3␣-120.0
H5␣1␣1.0890␣2␣120.0␣3␣120.0
H6␣2␣1.0890␣1␣109.4␣3␣180.0
End␣of␣z-Matrix
&seward
&scf
&grid_it
```

Now we write:

```
export Project=ethanol
molcas ethanol.input -o ethanol.out -e ethanol.err
```

Notice the addition of GRID_IT to obtain the graphical interface file ***ethanol.grid***.

3.6.3 Running a geometry optimization

In the next example we run a DFT/B3LYP geometry optimization of the ground state of the water molecule. Notice that, after &gateway has defined the coordinates and basis set definition, the EMIL commands `>>> Do while` and `>>> EndDo` are employed to form a loop

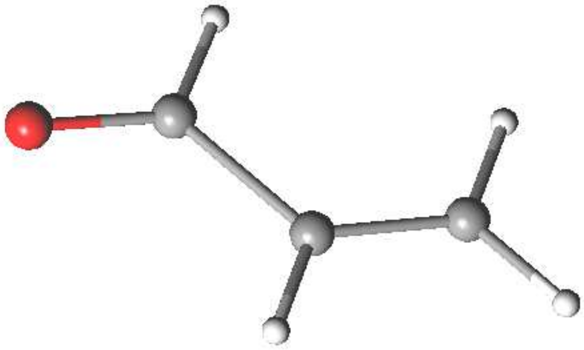


Figure 3.1: *The acrolein molecule.*

with the following programs util convergence of the geometry optimization: `SEWARD`, which computes the integrals in atomic basis, `SCF`, which, in this case, computes the DFT energy, and the program `SLAPAF` which controls the geometry optimization. It uses the `ALASKA` to compute the gradients of the energy with respect to the degrees of freedom. `SLAPAF` generates the new geometry to continue the iterative structure optimization process and check if the convergence parameters are satisfied. If they are, *MOLCAS* will be notified and the loop will stop.

```
&gateway
coord=water.xyz
Basis=ANO-S-MB
>>␣Do␣While
&seward
&scf
ksdft=b3lyp
&slapaf
>>>␣EndDo
```

Obviously this is just the default situation: optimizing a minimum without any further constraint. If other options are required (e.g., to get a transition state, a states crossing, to impose a geometry constraint, etc) specific input should be added to program `SLAPAF`.

One of the most powerful aspects of *MOLCAS* is the possibility of computing excited states with multiconfigurational approaches. The next example contains the calculation of the fifth lowest singlet roots in a State-Average (SA) CASSCF calculation using the `RASSCF` program, the addition of the dynamical correlation to get accurate electronic energies at the CASPT2 level, and the calculation of state-interaction properties such as oscillator strengths and others using the `RASSI` module.

```
&gateway
Coord
␣8
Angstrom
␣O␣␣␣␣␣-1.808864␣␣␣-0.137998␣␣␣␣0.000000
␣C␣␣␣␣␣␣1.769114␣␣␣␣0.136549␣␣␣␣0.000000
␣C␣␣␣␣␣␣0.588145␣␣␣-0.434423␣␣␣␣0.000000
␣C␣␣␣␣␣-0.695203␣␣␣␣0.361447␣␣␣␣0.000000
␣H␣␣␣␣␣-0.548852␣␣␣␣1.455362␣␣␣␣0.000000
␣H␣␣␣␣␣␣0.477859␣␣␣-1.512556␣␣␣␣0.000000
␣H␣␣␣␣␣␣2.688665␣␣␣-0.434186␣␣␣␣0.000000
␣H␣␣␣␣␣␣1.880903␣␣␣␣1.213924␣␣␣␣0.000000
```

```
Basis=ano-s-mb
Group=Nosym
&seward
title=acrolein
&rasscf
nactel=6␣0␣0
inactive=12
ras2=5
ciroot=5␣5␣1
&caspt2
multistate=5␣1␣2␣3␣4␣5
&rassi
Nr␣of␣Job=1␣5;␣1␣2␣3␣4␣5
EJob
```

Notice that the Group with the option Nosym has been used to prevent `GATEWAY` to identify the symmetry of the molecule (C_s in this case). Otherwise the input of the `RASSCF` program will have to change to incorporate the classification of the active space into the corresponding symmetry species. We skip working with symmetry at this stage, although its use is very convenient in many cases. A good strategy is to run only `GATEWAY` and let the program guide you.

The `RASSCF` input describes the active space employed, composed by six active electrons distributed in five active orbitals. By indicating twelve inactive orbitals (always doubly occupied) we complete the information about the total number of electrons and the distribution of the orbitals. Five roots will be obtained in the SA-CASSCF procedure and all them will be computed at the CASPT2 level to get the transition energies at the higher level. Further, the `RASSI` will compute the transition properties, in particular the transition dipole moments and the oscillator strengths.

3.6.4 Solvation effects

MOLCAS incorporates the effects of the solvent using several models. The most common is the cavity-based reaction-field Polarizable Continuum Model (PCM). It is incorporated by adding the keyword RF-input to the `SEWARD` code, in order to have the proper integrals computed.

```
&gateway
coord=CH4.xyz
Basis=ANO-S-MB
&seward
RF-Input
␣PCM-Model
␣Solvent=Water
End␣of␣RF-Input
&rasscf
Nactel=8␣0␣0
Inactive=1
Ras2=8
&caspt2
␣rfpert
```

The reaction field is computed in a self-consistent manner by the `SCF` or `RASSCF` codes, and added as a perturbation to the Hamiltonian in the other methods with they keyword RFPErt.

3.7 Basic and More Common Keywords

3.7.1 Environment and EMIL commands

Variable	*Purpose*
MOLCAS	MOLCAS home directory.
MOLCASMEM	Memory definition in Mb. Default 512.
MOLCAS_PRINT	Printing level: 2 Normal, 3 Verbose
Project	Name used for the project/files.
WorkDir	Scratch directory for intermediate files.

Command	*Purpose*
>>Do While	Start of a loop in an input file for geometry optimization with conditional termination.
>>Foreach	Start of a loop in an input file over a number of items.
>>EndDo	End of a loop in an input file.
>>If (condition)	Start of If block.
>>EndIf	End of If block.
>>Label Mark	Setting the label ”Mark” in the input.
>>Goto Mark	Forward jump to the label ”Mark” skipping that part of the input.

3.7.2 GATEWAY - Definition of geometry, basis sets, and symmetry

Keyword	*Meaning*
GROUP	Full (find maximum), Nosym, or generators (see Symmetry)
SYMMETRY	Symmetry generators: X, Y, Z, XY, XZ, YZ, XYZ
BASIS SET	Atom_label.Basis_label (for example ANO-L-VTZP)
COORD	File name or inline number of atoms and XYZ coordinates
RP-C	TS optimization via the Saddle method

RF-Input	Start of the Reaction field input for PCM Model. Water is the default solvent.
End of RF-Input	End of Reaction field input for PCM Model.
RICD	On-the-fly auxiliary basis sets.

3.7.3 SEWARD - An integral generation program

Keyword	*Meaning*
Choleski	Use Choleski decomposition
AMFI	Atomic mean-field integrals for relativistic calculations. Required for spin-coupling. Automatic for ANO-RCC basis sets

3.7.4 SCF - An SCF and DFT program

Keyword	*Meaning*
UHF	Unrestricted Hartee Fock calculation
KSDFt	DFT calculations, with options: BLYP, B3LYP, B3LYP5, HFB, HFS, LDA, LDA5, LSDA, LSDA5, SVWN, SVWN5, TLYP, PBE, PBE0
CHARge	Net charge of the system (default zero)
ZSPIn	Difference between α and β electrons

3.7.5 RASSCF - A multiconfigurational MCSCF program

Keyword	*Meaning*
SYMMetry	Symmetry of the wave function (according to `GATEWAY`) (1 to 8)
SPIN	Spin multiplicity

NACTEL	Three numbers: Total number of active electrons, holes in Ras1, particles in Ras3
INACTIVE	By symmetry: doubly occupied orbitals
RAS1	By symmetry: Orbitals in space Ras1 (RASSCF)
RAS2	By symmetry: Orbitals in space Ras1 (CASSCF and RASSCF)
RAS3	By symmetry: Orbitals in space Ras1 (RASSCF)
CIROOT	Three numbers: number of CI roots, dimension of the CI matrix, relative weights (typically 1)

3.7.6 CASPT2 - A second-order multiconfigurational perturbation program

Keyword	*Meaning*
MULTISTATE	Multistate CASPT2 calculation: number of roots and roots (Ex. 3 1 2 3)
FROZEN	By symmetry: non-correlated orbitals (default: core)
IMAG	Value for the imaginary shift for the zero order Hamiltonian

3.7.7 RASSI - A state-interaction properties program

Keyword	*Meaning*
NROFJOB	Number of input files, number of roots, and roots for each file
EJOB/HDIAG	Read energies from input file / inline
SPIN	Compute spin-orbit matrix elements for spin properties

3.7.8 MCLR program - A multiconfigurational linear-response program

Keyword	*Meaning*
SALA	Root to relax in geometry optimizations
ITER	Number of iterations

3.7.9 SLAPAF - A geometry-relaxation program

Keyword	*Meaning*
TS	Computing a transition state
FINDTS	Computing a transition state with a constraint
MEP-SEARCH	Computing a steepest-descent minimum reaction path
ITER	Number of iterations
CONSTRAINTS	Definition of the geometrical or energetic constraints
INTERNAL	Definition of the internal coordinates
IRC	Intrinsic reaction coordinate analysis of a TS

3.7.10 GRID_IT - A program to compute densities and graphical interface

Keyword	*Meaning*
ASCII	Generate the `grid` file in ASCII

3.7.11 MBPT2 - An MP2 program

Keyword	*Meaning*
FROZEN	By symmetry: non-correlated orbitals (default: core)

3.7.12 MOTRA - A molecular-basis transformation program

Keyword	*Meaning*
FROZEN	By symmetry: non-correlated orbitals (default: core)
RFPERT	Previous reaction field introduced as a perturbation
LUMORB	Input orbital file as ASCII (INPORB)
JOBIPH	Input orbital file as binary (JOBOLD)

3.7.13 CCSDT - A coupled-cluster program

Keyword	*Meaning*
CCSD	Coupled-cluster singles and doubles method
CCT	CCSD plus a non iterative triples (T) calculation

3.8 Getting Information: Outputs and GUI

MOLCAS provides a lot of printed information in the outputs of the program. The printing level is controlled by the environmental variable MOLCAS_PRINT, which by default is set to two. To increase the printing level the variable can be also increased for instance to three. A typical *MOLCAS* output contains the program header and input information, the conditions of the calculation, the steps to obtain or converge the energies and wave functions, and the final results, including in many cases the molecular orbital coefficients and the analysis of the properties for the computed states.

For geometry optimizations where many steps are required and a lot of output is generated there are different options to control its generation. Three EMIL commands can be used: Set Output Screen redirects the output to the screen; Set Output Over, which is the default, skips the outputs of the intermediate steps and only produces the output of the last iteration, and Set Output File places all outputs from each of the iterations in the $WorkDir directory in the files named ***`Structure.iter.output`***, where iter is the number of the iteration. This is a convenient option to follow closely the convergence process. In this case the MOLCAS_PRINT command must be set to three.

3.8.1 MING: a Graphical Molcas Input Generator

MOLCAS has incorporated a graphical self-guided tool to help the user to prepare the inputs and calculation flow named `MING`. Provided that your system has all the graphical libraries and system utilities required for the code and this has been properly installed (try configure -ming and read the installation guide if something fails), the `MING` program is activated by the command molcas ming.

A window will then open in which the left panel contains three entrances. Clicking on TEMPLATES selected types of calculation are displayed in the right panel with the proper flow of *MOLCAS* modules. Pressing on each of such boxes open new windows that enables to fill the input of the program. Most windows have a basic (default) and an advanced form. New modules or commands can be added to the flow by using the two other entrance in the left panel: COMMANDS, including the EMIL commands that control the flow or add specific information, and MODULES, including all *MOLCAS* programs and utilities (see below).

In the upper toolbar we find Settings, to define the *MOLCAS* environment, tools to Add, Duplicate, Delete or Clear new entrances, Preview and Edit the prepared input files, command

Open to retrieve previous input files, saving commands, and even commands to Submit to send the calculation and command Watch to inspect the output and error files.

`MING` can prepare most of calculations available in MOLCAS. Even if you have a complex calculation the tool can be used to simplify and speed the basic aspects of the input.

3.8.2 GV: Grid an Geometry visualization

MOLCAS has developed a graphical interface that can be used both to create inputs for the program and especially to analyze the results in a graphical way, in particular visualization of molecular orbitals, density plots, etc. The program is invoked with the general command molcas gv [-option value] file1, and it has the following uses:

- molcas gv xyz_file: coordinate mode reading from a cartesian coordinates file. You can see and modify the molecule with the use of the left-button of the mouse and the keyboard. Here there are some of the most useful commands. Get the others with the molcas help gv utility.

Left mouse click:	Select atoms (if two, a bond is selected, if three a bond angle, if four a dihedral angle
Left mouse + Shift click:	Mark/unmark atoms to/from the group
Middle mouse/Space:	Remove selection, or marking
F3 key:	Open fragment menu
Insert key:	Insert atom
PageUp, PageDown:	Alter type of selected atom or bond
Delete/Supress key:	Delete a selected atom
+/-:	Change a value of selected bond/angle in steps
F4 key:	Input window. Type (+ Enter) a value of selected bond/angle
Backspace:	Undo last action
Home:	Set selected atom to center of coordinates
F8 key:	Find or apply symmetry
F2 key:	Save XYZ or INPORB
F1 key:	Get help on screen
Right mouse click:	Options for colors, display, and saving

- molcas gv molden_file: also coordinate mode, reading from a MOLDEN file.

- molcas gv -c molcas_input_file: coordinate mode, reading from a MOLCAS `GATEWAY/SEWARD` input.

- molcas gv grid_file: coordinates and density mode, reading from a binary/ascii *`grid_file`*. This file is generated by `GRID_IT` and by default placed in the *`$WorkDir`* directory with the name *`$Project.grid`*. The program allows displaying total densities, molecular orbitals, and charge density differences.

Right mouse click:	A menu that allows to select the orbital, change its character (frozen, inactive, Ras1, Ras2, Ras3, secondary), change display and color options, and different save options for orbitals and graphical files
F3 key:	All available molecular orbitals displayed on the screen. The different background color indicates the orbital space to which they belong
Left mouse click:	Rotate orbitals individually, even in the multiview option
Cursor arrows:	Displace the orbital
PageUp, PageDown:	Change grid/orbital
+/-:	Change isolevel/size of the orbital

Section 4

Tutorials

4.1 Problem-based tutorials

4.1.1 Computing the electronic energy at fixed nuclear geometry

The *MOLCAS* 7.4 suite of quantum chemical programs is modular in design. The desired calculation is achieved by executing a list of *MOLCAS* program modules in succession, occasionally manipulating the program information files. If the information files from a previous calculation are saved then a subsequent calculation need not recompute them. This is dependent on the correct information being preserved in the information files for the subsequent calculations. Each module has keywords to specify thefunctions to be carried out and many modules rely on the specification of keywords in previous modules.

In the present examples the calculations will be designed by preparing a single file in which the input for the different programs is presented sequentially. Our initial problem will be to compute an electronic energy at a fixed geometry of the nuclei, and this will be done using different methods and thus different program modules.

First, the proper *MOLCAS* environment has to be set up. The following variables must be properly defined, for instance:

```
export␣MOLCAS=/home/molcas/molcas.7.4
export␣Project=CH4
export␣WorkDir=/home/user/tmp
```

If not defined, *MOLCAS* gives them the default values. The MOLCAS variable will be set up as the latest implemented version of the code. Once in the *MOLCAS* home directory the variable is set up directly. Project and WorkDir have the default values None and $PWD, respectively. It is very important that the molcas driver, called by command `molcas` and built during the installation of the code, is included in the $PATH.

We prepare our first run: the calculation of the SCF energy of the methane (CH_4) molecule. Three programs should be used: `GATEWAY` to specify information about the system, `SEWARD` to compute and store the one- and two-electron integrals, and `SCF` to obtain the Hartree-Fock SCF wave function and energy. Three inputs should be prepared. The `GATEWAY` input will contain the nuclear geometry in cartesian coordinates and the label for the one-electron basis set, here, for instance, an STO-3G minimal basis set. We shall use here the keyword

COORD, which allows to automatically create a `GATEWAY` input from a standard file containing the cartesian coordinates in Angstrom (as generated by many programs like `MOLCAS GV` or `MOLDEN`). No symmetry is being considered so far, therefore we introduce the keyword NOSYM to force the program not to look for any symmetry in the molecule. In this case the input for `SEWARD` is void. In a closed-shell case like this one the `SCF` input can be left empty. All the input files used here can be found at $MOLCAS/doc/examples/tutorials, like the file *`SCF.energy.CH4`* described below.

```
*SCF␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry.
*File:␣SCF.energy.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&SCF
Title␣=␣CH4␣molecule
```

where the content of the *`CH4.xyz`* file is:

```
5
Angstrom
C␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣0.000000
H␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣1.050000
H␣␣␣␣1.037090␣␣␣␣␣0.000000␣␣␣␣-0.366667
H␣␣␣-0.542115␣␣␣␣-0.938971␣␣␣␣-0.383333
H␣␣␣-0.565685␣␣␣␣␣0.979796␣␣␣␣-0.400000
```

Now, to run *MOLCAS* we simply type:

```
molcas␣SCF.energy.CH4.input␣>␣SCF.energy.CH4.out␣2␣>␣SCF.energy.CH4.err
or,
molcas␣-f␣SCF.energy.CH4.input
```

and the main output will be stored in file *`SCF.energy.CH4.out`* (or in the second case *`SCF.energy.CH4.log`*: and the default error file in *`SCF.energy.CH4.err`*. The most relevant information will be contained in the output file, where we shall analyze the `SEWARD` program information, describing the nuclear geometry, molecular symmetry, and the data regarding the one-electron basis sets and the calculation of one- and two-electron integrals, as described in section 4.2.2. Next, comes the output of program `SCF` with information of the electronic energy, wave function, and the Hartree-Fock (HF) molecular orbitals (see section 4.2.3). Files containing intermediate information, integrals, orbitals, etc, will be kept in the $WorkDir directory for further use. For instance files *`$Project.OneInt`* and *`$Project.OrdInt`* contains the one- and two-electron integrals stored in binary format. File *`$Project.ScfOrb`* keeps the HF molecular orbitals in ASCII format. Finally, *`$Project.RunFile`* is a communication file between programs. All those files can be later used for more advanced calculations in order to avoid repeating the already run calculations.

There are some graphical utilities that can be used for the analysis of the results. By default, *MOLCAS* generates files which can be read with the `MOLDEN` program. In particular we can find in $WorkDir the file *`CH4.scf.molden`*, where information about molecular geometry, molecular orbitals (use Density Mode in `MOLDEN`), etc, can be found (See also section **??**). However, *MOLCAS* has its own graphical tool, program `GV`, which is an openGL based viewer and allows to visualize molecular geometries, orbitals, densities, and density differences. For instance, to obtain a graphical display of the molecule from a standard coordinate file we type:

```
molcas␣gv␣CH4.xyz
```

while option gv -c should be used if reading from a `SEWARD` input. Once in the screen the molecule can be manipulated in many ways, fo example, changing its geometry, adding or deleting atoms, etc. You can also use it to build new molecules.

In order to obtain the information to display molecular orbitals and densities it is necessary to run a *MOLCAS* program named `GRID_IT`

```
*SCF␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry␣plus␣a␣grid␣for␣visualization.
*File:␣SCF.energy_grid.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
Group␣=␣C1
␣&SEWARD;␣&SCF
␣&GRID_IT
All;␣Ascii
```

and run it:

```
molcas␣SCF.energy_grid.CH4.input␣>␣SCF.energy_grid.CH4.out␣2␣>␣SCF.energy_grid.CH4.err
```

`GRID_IT` can also be run separately if an orbital file is specified in the input and the $WorkDir directory is available. In the $WorkDir and $PWD directories a new file is generated, *`CH4.grid`* which contains the information required by the `GRID_IT` input. By typing:

```
molcas␣gv␣CH4.grid
```

a window will be opened displaying the molecule and its charge density. By proper selection with the mouse buttons, shape and size of several molecular orbitals can be selected. Typing molcas gv -? one can get a short help for the use of the program. More information can be found in sections 8.16 for `GRID_IT` and 9.1 `GV` programs.

As an alternative to run the projects we include here a short script to be placed in the directory $MOLCAS/doc/samples/problem_based_tutorials as the file *`project.sh`*. Just run project.sh $Project, where $Project is the MOLCAS input and you will get output and error files and a $WorkDir directory named $Project.work.

```
#!/bin/bash
cd␣../../..
export␣MOLCAS=$PWD
export␣MOLCASDISK=2000
export␣MOLCASMEM=64
export␣MOLCAS_PRINT=3
export␣Project=$1
export␣HomeDir=$MOLCAS/doc/samples/problem_based_tutorials
export␣WorkDir=$HomeDir/$Project.work
mkdir␣$WorkDir␣2>/dev/null
cp␣$HomeDir/*xyz␣$WorkDir/
cd␣$WorkDir
```

```
molcas␣$HomeDir/$1␣>$HomeDir/$Project.out␣2>$HomeDir/$Project.err
exit
```

In order to run a Kohn-Sham density functional calculation, MOLCAS uses the same SCF module, therefore the only change needed is to specify in the SCF input the DFT option and the required functional, for instance here, B3LYP:

```
*DFT␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry␣plus␣a␣grid␣for␣visualization.
*File:␣DFT.energy.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&SCF
Title␣=␣CH4␣molecule
KSDFT␣=␣B3LYP
␣&GRID_IT
All
```

and the same graphical files can be found in $WorkDir and $PWD.

Our next step is to get the second-order Møller–Plesset perturbation (MP2) energy for methane at the same molecular geometry and using the same one-electron basis set. Program MBPT2 has to be invoked this time. We can take advantage of having previously computed the proper integrals with SEWARD and the reference closed-shell HF wave function with the SCF program. In such case we may keep the same definitions as before and simply prepare a file containing the MBPT2 input and run it using the molcas command. The proper intermediate file will be already in $WorkDir. If, on the contrary, one has to start from scratch, all required inputs should be placed sequentially, for instance, in the *MP2.energy.CH4* file. If we decide to start the project from the beginning is convenient to remove the $WorkDir directory, unless we are certain about the files we are going to find there.

```
*MP2␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry.
*File:␣MP2.energy.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&SCF
␣&MBPT2
Title␣=␣CH4␣molecule
Frozen␣=␣1
```

On top of the HF wave function, an MP2 calculation has been performed with the deepest orbital, carbon 1s, of the molecule frozen. Information about the output of the MBPT2 program can be found on section 4.2.13.

The SCF program works by default with closed-shell systems with an even number of electrons at the Restricted Hartee-Fock (RHF) level. We may instead want to use the Unrestricted Hartree Fock (UHF) method, by invoking the keyword UHF. This is possible for both even and odd electron systems. For instance in a system with odd number of electrons like the CH$_3$ radical

```
4
Angstrom
C␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣0.000000
H␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣1.050000
H␣␣␣␣1.037090␣␣␣␣␣0.000000␣␣␣␣-0.366667
H␣␣␣-0.542115␣␣␣␣-0.938971␣␣␣␣-0.383333
```

the input to run an open-shell UHF calculation will be simply

```
*SCF/UHF␣energy␣for␣CH3␣at␣a␣fixed␣nuclear␣geometry
*File:␣SCF.energy_UHF.CH3
*
␣&GATEWAY
Title␣=␣CH3␣molecule
coord␣=␣CH3.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&SCF
Title=␣CH3␣molecule
UHF
```

If we want to force a charged system, this must be indicated in the `SCF` input, for instance to compute the cation of the CH_4 molecule at the UHF level:

```
*SCF/UHF␣energy␣for␣CH4+␣at␣a␣fixed␣nuclear␣geometry
*File:␣SCF.energy_UHF.CH4plus
*
␣&GATEWAY
Title␣=␣CH4+␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣c1
␣&SEWARD
␣&SCF
Title␣=␣CH4+␣molecule
UHF
Charge␣=␣+1
```

The Kohn-Sham DFT calculation can be also run on the basis of the UHF algorithm

```
*DFT/UHF␣energy␣for␣CH4+␣at␣a␣fixed␣nuclear␣geometry
*File:␣DFT.energy.CH4plus
*
␣&GATEWAY
Title␣=␣CH4+␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&SCF
Title␣=␣CH4+␣molecule
KSDFT␣=␣B3LYP
UHF
Charge␣=␣+1
```

In the UHF (and UHF/DFT) method it is also possible to force particular α and β orbital occupations. There are two ways to do it. First, with keyword ZSPIN in the `SCF` program, which represents the difference between the number of α and β electrons. For instance, we can set the keyword to two and force the program to converge a situation with two more α than β electrons.

```
*DFT/UHF␣energy␣for␣different␣electronic␣occupation␣in␣CH4␣at␣a␣fixed␣nuclear
geometry
*File:␣DFT.energy_zspin.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣c1
␣&SEWARD
␣&SCF
Title␣=␣CH4␣molecule␣zspin␣2
UHF;␣ZSPIN␣=␣␣2
KSDFT␣=␣␣B3LYP
```

In the output the final occupations are six α and four β orbitals. Alternatively, instead of ZSpin we may force from the beginning the occupation numbers with keyword Occupation, followed by one line containing the occupied α orbitals, six here, and another line with the β orbitals, four here. Sometimes convergence can be improved with this option.

Different sets of methods use other *MOLCAS* modules. For instance, to perform a Complete Active Space (CAS) SCF calculation, the `RASSCF` program has to be used. This module requires starting trial orbitals, which can be obtained from a previous SCF calculation or automatically by `SEWARD`, which provides trial orbitals by using a model Fock operator. Recommended keywords are Nactel, defining the total number of active electrons, holes in Ras1, and particles in Ras3, respectively (the two last ones only for RASSCF-type calculations), Inactive, including the number of orbitals inactive (occupation always two) in the CASSCF reference, and Ras2, defining the number of active orbitals. By default the obtained wave function will correspond to the lowest state of the symmetry with spin multiplicity one. Most of the input can be avoided if one has prepared and linked an INPORB file with the different orbital types defined (for example with the programs `GV`).

```
*CASSCF␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣CASSCF.energy.CH4
*
␣&GATEWAY
coord␣=␣CH4.xyz
basis␣=␣STO-3G
group␣=␣C1
␣&SEWARD
␣&RASSCF
␣␣␣Title␣=␣CH4␣molecule
␣␣␣Spin␣=␣1;␣Nactel␣=␣8␣0␣0;␣Inactive␣=␣1;␣Ras2␣=␣8
␣&GRID_IT
All;␣Ascii
```

In this case we are computing the lowest singlet state, the ground state here, considering that this is a closed-shell situation, with an active space of eight electrons in eight orbitals. The lowest orbital of the molecule (carbon 1s) has been left inactive. This is a CASSCF case in which all the valence orbitals and electrons (carbon 2s2p plus four hydrogen 1s) have been included in the active space.

Using the CASSCF wave function as a reference, we can perform a second-order perturbative, CASPT2, correction to the electronic energy by employing the `CASPT2` program. If we maintain in the $WorkDir directory the files generated by the previous run (in particular the following files: integral files *CH4.OneInt*,*CH4.OrdInt*; CASSCF wave function information file *CH4.JobIph*, and communication file *CH4.RunFile*), it will not be necessary to run again programs `SEWARD`, and `RASSCF`. In that case it will be enough to prepare an input for the `CASPT2` program and run it. Here, however, we include the full input file:

```
*CASPT2␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣CASPT2.energy.CH4
*
␣&GATEWAY
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣C1
␣&SEWARD
␣&RASSCF
␣␣␣LumOrb
␣␣␣Title␣=␣CH4␣molecule
␣␣␣Spin␣=␣1;␣Nactel␣=␣6␣0␣0;␣Inactive␣=␣2;␣Ras2␣=␣6
␣&CASPT2
Title␣=␣CH4␣molecule
Multistate␣=␣1␣1
```

As an alternative to generate trial orbitals for the `RASSCF` program, we have added here an input for the `SCF` program. In most of cases the Hartree-Fock orbitals will be a better choice as starting orbitals. In that case, the `RASSCF` input has to include keyword LumOrb to read from any external source of orbitals other than those generated by the `SEWARD` program. We have also changed the input of the `RASSCF` program. Now the number of active orbitals is six, as well as the active electrons. We have excluded from the space the two lowest orbitals (Inactive 2) and also another orbital goes to the secondary space. If we had stayed with the previous (8,8) full valence space, considering that we are using a minimal basis set, the `CASPT2` program would not have more electronic correlation energy to include. Regarding the input for the `CASPT2` program we have frozen the lowest orbital (carbon 1s).

We may change the charge and multiplicity of our wave function by computing the CH_4^+ cation with the same methods. It is the `RASSCF` program which defines the character of the problem by specifying number of electrons, and spin and spatial symmetry. We have here one electron less and a doublet multiplicity.

```
*CASSCF␣energy␣for␣CH4+␣at␣a␣fixed␣nuclear␣geometry
*File:␣CASSCF.energy.CH4plus
*
␣&GATEWAY
Title␣=␣CH4+␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣Group␣=␣C1
␣&SEWARD
␣&RASSCF
␣␣␣Title␣=␣CH4+␣molecule
␣␣␣Spin␣=␣2;␣Nactel␣=␣5␣0␣0;␣Inactive␣=␣2;␣Ras2␣=␣6
```

No further modification is needed in the `CASPT2` input:

```
*CASPT2␣energy␣for␣CH4+␣at␣a␣fixed␣nuclear␣geometry
*File:␣CASPT2.energy.CH4plus
*
␣&GATEWAY
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣C1
␣&SEWARD
␣&RASSCF
␣␣␣Title␣=␣CH4+␣molecule
␣␣␣Spin␣=␣2;␣Nactel␣=␣1␣0␣0;␣Inactive␣=␣4;␣Ras2␣=␣1
␣&CASPT2
Title␣=␣CH4␣molecule
```

A somewhat more sophisticated calculation can be performed at the Restricted Active Space

(RAS) SCF level. In such case the level of excitation in the CI expansion can be controlled by restricting the number of holes and particles present in certain orbitals.

```
*RASSCF␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣RASSCF.energy.CH4
*
␣&GATEWAY
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣C1
␣&SEWARD
␣&RASSCF
␣␣␣Title␣=␣CH4␣molecule
␣␣␣Spin␣=␣1;␣Nactel␣=␣8␣1␣1
␣␣␣Inactive␣=␣1;␣Ras1␣=␣1;␣Ras2␣=␣6;␣Ras3␣=␣1
```

In particular the previous calculation includes one orbital within the Ras1 space and one orbital within the Ras3 space. One hole (single excitation) at maximum is allowed from Ras1 to Ras2 or Ras3, while a maximum of one particle is allowed in Ras3, either coming from Ras1 or Ras2. Within Ras2 all type of orbital occupations are allowed. The RASSCF wave functions can, in principle, be used as reference for multiconfigurational perturbation theory (RASPT2) but this approach is still in the experimental stage.

MOLCAS also has the possibility of computing electronic energies at different CI levels by using the `MRCI` program. We have prepared here a Singles and Doubles CI (SDCI) calculation on the CH_4 molecule. To set up the calculations, programs `MOTRA`, which transforms the integrals to molecular basis, and `GUGA`, which computes the coupling coefficients, have to be run before the `MRCI` program. In `MOTRA` we have specified the reference orbitals employed (those from an HF `SCF` calculation) and the frozen orbitals. In `GUGA` we describe the reference for the CI, with the number of correlated electrons, the spatial and spin symmetry, the inactive orbitals (always occupation two in the reference space), and the type of CI expansion.

```
*SDCI␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣SDCI.energy.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD
␣&SCF
Title=␣CH4␣molecule
␣␣&MOTRA
Lumorb
Title=␣CH4␣molecule
Frozen=␣1
␣␣&GUGA
Title=␣CH4␣molecule
Electrons␣=␣8
Spin␣=␣1
Inactive=␣4
Active=␣0
Ciall=␣1
␣␣&MRCI
Title=␣CH4␣molecule
SDCI
```

If we want to use reference orbitals from a previous CASSCF calculation, the `RASSCF` program will have to be run before the `MOTRA` module. Also, if the spatial or spin symmetry are changed for the CI calculation, the modifications will be introduced in the `GUGA` input. Many alternatives are possible in order to perform an MRCI calculation like next one, in which the reference space to perform the CI is multiconfigurational:

```
*MRCI␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣MRCI.energy.CH4
*
␣&GATEWAY
Title␣=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD;␣&SCF
␣&RASSCF
LumOrb
Title=␣CH4␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣2;␣Ras2=␣6
␣␣&MOTRA
Lumorb
Title=␣CH4␣molecule
Frozen=␣1
␣␣&GUGA
Title=␣CH4␣molecule
Electrons=␣8
Spin=␣1
Inactive=␣2
Active=␣3
Ciall=␣1
␣␣&MRCI
Title=␣CH4␣molecule
SDCI
```

The `MRCI` program allows also to get electronic energies using the ACPF method. Another *MOLCAS* program, `CPF`, offers the possibility to use the CPF, MCPF, and ACPF methods with a single reference function. The required input is quite similar than that for the `MRCI` program:

```
*CPF␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣CPF.energy.CH4
*
␣&GATEWAY
Title=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD;␣&SCF
␣&MOTRA
Lumorb
Title=␣CH4␣molecule
Frozen=␣1
␣␣&GUGA
Title=␣CH4␣molecule
Electrons=␣8
Spin␣=␣1
Inactive␣=␣4
Active␣=␣0
Ciall=␣1
␣␣&CPF
Title=␣CH4␣molecule
CPF
```

Finally, *MOLCAS* can also perform closed- and open-shell coupled cluster calculations at the CCSD and CCSD(T) levels. They are controlled by the `CCSDT` program, whose main requirement is that the reference function has to be generated with `RASSCF`. The following input is required to obtain the CCSD(T) energy for the CH_4 molecule:

```
*CCSDT␣energy␣for␣CH4␣at␣a␣fixed␣nuclear␣geometry
*File:␣CCSDT.energy.CH4
*
␣&GATEWAY
```

```
Title=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD
␣&RASSCF
␣␣␣Title=␣CH4␣molecule
␣␣␣Spin=␣1;␣Nactel=␣0␣0␣0;␣Inactive=␣5;␣Ras2=␣0
␣␣␣OutOrbitals
␣␣␣Canonical
␣&MOTRA
JobIph
Title=␣CH4␣molecule
Frozen=␣1
␣&CCSDT
Title=␣CH4␣molecule
CCT
```

As it is a closed-shell calculation, the `RASSCF` input is prepared to compute a simple RHF wave function (zero active electrons and orbitals), with keywords OutOrbitals Canonical. The `MOTRA` has to include the keyword JobIph to extract the wave function information from file *`JOBIPH`* (automatically generated by `RASSCF`). Finally, CCT in program `CCSDT` leads to the calculation of the CCSD(T) energy using the default algorithms.

The `CCSDT` program in MOLCAS is specially suited to compute open-shell cases. The input required to obtain the electronic energy of the CH_4^+ cation with the CCSD(T) method is:

```
*CCSDT␣energy␣for␣CH4+␣at␣a␣fixed␣nuclear␣geometry
*File:␣CCSDT.energy.CH4plus
*
␣&GATEWAY
Title=␣CH4+␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD
␣&RASSCF
Title=␣CH4+␣molecule
Spin=␣2;␣Nactel=␣1␣0␣0;␣Inactive=␣4;␣Ras2=␣1
OutOrbitals
␣Canonical
␣&MOTRA
JobIph
Title=␣CH4+␣molecule
Frozen=␣1
␣&CCSDT
Title=␣CH4+␣molecule
CCT
```

where it is the `RASSCF` program which generated the proper Restricted Open-Shell (RO) HF reference. Different levels of spin adaptation are available.

We may want to introduce the effect of solvents into our calculation. MOLCAS includes two models: Kirkwood and PCM. To add solvent effect to a ground state at the HF, DFT, or CASSCF levels simply requires to include the RF-input within `SEWARD`, while the reaction field is computed self-consistently.

```
*DFT␣energy␣for␣CH4␣in␣water␣at␣a␣fixed␣nuclear␣geometry
*File:␣DFT.energy_solvent.CH4
*
␣&GATEWAY
Title=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD
RF-input
```

```
␣␣PCM-model;␣solvent=␣water
End␣of␣RF-input
␣&SCF
Title=␣CH4␣molecule
KSDFT=␣B3LYP
```

Other programs such as `CASPT2`, `RASSI`, and `MOTRA` require that the reaction field is included as a perturbation with keyword RFPERTURBATION. In the next example the correction is added at both the CASSCF and CASPT2 levels.

```
*CASPT2␣energy␣for␣CH4␣in␣acetone␣at␣a␣fixed␣nuclear␣geometry
*File:␣CASPT2.energy_solvent.CH4
*
␣&GATEWAY
Title=␣CH4␣molecule
coord␣=␣CH4.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD
RF-input
␣␣␣PCM-model;␣solvent=␣acetone;␣AAre=␣0.2
End␣of␣RF-input
␣&RASSCF
Title=␣CH4␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣2;␣Ras2=␣6
␣&CASPT2
Title=␣CH4␣molecule
Frozen=␣1
Multistate=␣1␣1
RFPert
```

Notice that the tesserae of the average area in the PCM model (keyword AARE has been changed to the value required for acetone, while the default is 0.4 Å^2 for water (see section ??). More detailed examples can be found in section 5.6.

4.1.2 Optimizing geometries: minima, transition states, crossings, and minimum energy paths

We may want now to explore potential energy surfaces (PES) and optimize the molecular geometry for specific points in the PES. Different cases can be accomplished, like getting the optimal geometry in a minimum energy search , to obtain a transition state structure connecting different regions of the PES, to find the crossing between two PES in which the energy becomes degenerate, or mapping the minimum (steepest descendent) energy path (MEP) from a specific point if the PES downward to a minimum energy situation. All these searches can be performed fully optimizing all the degrees of freedom of the system or introducing certain restrictions. MOLCAS 7.4 can perform geometry optimizations at the SCF (RHF and UHF), DFT (RHF and UHF based), CASSCF (CASSCF and RASSCF), and CASPT2 and other correlated levels, although in the last case only numerical gradients are available, where in the other cases more efficient analytical gradients are used.

Geometry optimizations require many loops, in which the electronic energy is estimated at a specific level of calculation and the gradients of the energy (first derivatives) with respect to the geometric degrees of freedom (DOF) are computed. With this information at hand the program has to decide if we are already at the final required geometry (for instance if gradient is zero for all DOF we may have reached a minimum in the PES) or if we must change the geometry to try to get a better solution in a new iteration. The input file should therefore be built in a way that allows to loop over the different programs. The general input commands Do while and Enddo control the loop and the input is be inserted within them. We may want to add also instructions about the number of maximum iterations allowed and the type of output required (see section 8.9). We shall use here commands Set output file, which prints outputs for all iterations and places them in the $WorkDir directory with the name Structure.$iteration.output, and Set maxiter 100, which sets maximum iterations to one hundred.

The selected example is the geometry optimization of the water molecule at the SCF RHF level of calculation:

```
*SCF␣minimum␣energy␣optimization␣for␣H2O
*File:␣SCF.minimum_optimization.H2O
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>>␣Do␣while
␣&SEWARD␣;&SCF;␣&SLAPAF
>>>␣EndDo
```

The sequence of programs employed is `GATEWAY` (out of the loop), and then `SEWARD`, `SCF`, and `SLAPAF`. `SEWARD` is computes the integrals `SCF` program computes the RHF energy, and wave function. `SLAPAF` will control the calculation of gradientns and estimate if the calculation has already finished or need to proceed to a new nuclear geometry for the next iteration. Automatically a file named *`$Project.geo.molden`* will be generated in $WorkDir containing all the geometric steps contained in the optimization process. `MOLDEN` can read such file.

To move to another reference wave function means simply to change the initial programs. For instance, we can perform an UHF calculation of the H_2O^+ cation:

```
*UHF␣minimum␣energy␣optimization␣for␣H2O+
*File:␣UHF.minimum_optimization.H2Oplus
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>␣Do␣while
␣&SEWARD
␣&SCF;␣Title="H2O␣minimum␣optimization";␣UHF;␣Charge=1
␣&SLAPAF
>>␣EndDo
```

The same procedure can be followed if we pretend to perform a DFT geometry optimization:

```
*DFT␣minimum␣energy␣optimization␣for␣H2O
*File:␣DFT.minimum_optimization.H2O
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&SCF␣;␣Title="H2O␣minimum␣optimization";␣KSDFT=B3LYP
␣&SLAPAF␣&END
>>>␣EndDo
```

Once we have found an energy minimum based on the calculation of gradients, we may ensure that this is really a minimum energy point, what can be only accomplished by computing second derivatives of the energy, that is, the Hessian. MOLCAS can compute analytical Hessians for SCF and single state CASSCF wave functions. For some other methods one can use numerical procedures to compute the Hessian. Once the Hessian is computed we have the vibrational frequencies and we can apply statistical mechanics to obtain thermodynamic properties. If we are in a true energy minimum, all frequencies will be real (instead of imaginary). Program `MCKINLEY` computes second derivatives of a predefined (SCF or CASSCF) wave function, while `MCLR` performs the vibrational and statistical analyses. MOLCAS simply requires an input for the `MCKINLEY` program to do all the job by using keywords Perturbation Hessian, while program `MCLR` will be called automatically and no input is required. We include here the full set of calculations, first, a geometry optimization, followed by the calculation of a Hessian.

```
*SCF␣minimum␣energy␣optimization␣plus␣hessian␣of␣the␣water␣molecule
*File:␣SCF.minimization_plus_hessian.H2O
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&SCF;␣Title="H2O␣minimum␣optimization"
␣&SLAPAF␣&END
>>>␣EndDo
␣&MCKINLEY
```

Notice that `MCKINLEY` input is placed after EndDo, therefore, out of the looping scheme. Once the geometry optimization has finished, the Hessian will be computed at the final obtained geometry and computation level. In general, any calculation performed using a

$WorkDir directory where a previous geometry optimization of any type has taken place will use as geometry the last one obtained in the optimization, even if a `SEWARD` input is present. To avoid that, the only solution is to remove the communication file ***`RUNFILE`*** where the geometry is stored. Notice also that the frequencies are computed in a cartesian basis, and that three translational and three rotational frequencies (they should be very close to zero) are included in the output file (this is not the case when numerical gradients and Hessians are used). In particular, for water in its minimum energy structure we expect (3N-6) three real vibrational frequencies. By default in $WorkDir a file *`$Project.freq.molden`* is generated containing information about the vibrational frequencies, and modes, which can be visualized by `MOLDEN`.

We can introduce here a new level of theory, CASSCF, although this is especially suited for geometry optimizations of excited states (we shall consider excited states in the next chapter). A geometry optimization is performed, and to show more possibilities this time we shall impose the restriction that the HOH angle in water should be 120^o at the final structure. That means that only the bond distances will be really optimized in this partial minimization. The restriction is indicated in `SLAPAF` opening the keyword CONSTRAINTS, which ends with another keyword, END OF CONSTRAINTS. Inside we assign the name of some variables (one here) to the geometrical parameters we want to restrict (see nomenclature in section 8.34.4), in internal or cartesian coordinates. Here we want that the angle H1-O-H2 becomes finally 120^o, therefore we first assign as variable a the defined angle and after VALUE, variable a is determined as 120^o. It is not required that the initial geometry is 120^o, just that the final result for the calculation will become 120^o.

Observe also that as the `RASSCF` program requires initial trial orbitals, we use those which were automatically generated from `SEWARD`. The defined CASSCF wave function includes here all valence orbitals and electrons.

```
*CASSCF␣minimum␣energy␣optimization␣of␣the␣water␣molecule␣with␣geometrical␣restrictions
*File:␣CASSCF.minimum_optimization_restricted.H2O
␣&Gateway
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S-MB
O␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S-MB
H1␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
Constraint
␣␣␣a␣=␣Angle␣H1␣O␣H2
␣␣Value
␣␣␣a␣=␣90.␣degree
End␣of␣Constraints
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&RASSCF;␣nActEl=8␣0␣0;␣Inactive=1;␣Ras2=6
␣&SLAPAF
>>>␣EndDo
```

Other more flexible ways to impose geometric restrictions is to decide which internal coordinates should remain fixed and which should change. In the next example we force the bond lengths to remain fixed at their initial distance (here 0.91 Å), while the bond angle (initially 81°) is optimized.

```
*DFT␣minimum␣energy␣optimization␣of␣the␣angle␣in␣the␣water␣molecule␣at␣fixed␣bond␣lengths
*File:␣DFT.minimum_optimization_restricted.H2O
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S-MB
O␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S-MB
H1␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD;␣&SCF;␣Title="H2O␣restricted␣minimum";␣KSDFT=B3LYP
␣&SLAPAF
Internal␣Coordinates
␣␣␣b1␣=␣Bond␣O␣H1
␣␣␣b2␣=␣Bond␣O␣H2
␣␣␣a1␣=␣Angle␣H1␣O␣H2
␣␣Vary
␣␣␣a1
␣␣␣Fix
␣␣␣b1
␣␣␣b2
End␣of␣Internal
>>>␣EndDo
```

It the final output the bond lengths remain as initially while the angle is optimized to reach 112^o.

Our next step is to compute a transition state, a structure connecting different regions of the potential energy hypersurface which is a maximum just for one degree of freedom. The most common saddle points have order one, that is, they are maxima for one of one displacement and minima for the others. The simplest way to search for a transition state in *MOLCAS* is to add keyword TS to the `SLAPAF` input. Keyword PRFC is suggested in order to verify the nature of the transition structure. Searching for transition states is, however, not an easy task. Here we illustrate it for water at the DFT level:

```
*DFT␣transition␣state␣optimization␣of␣the␣water␣molecule
*File:␣DFT.transition_state.H2O
*
␣&Gateway
Title=␣H2O␣TS␣optimization
Basis␣set
O.ANO-S...3s2p.
O␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.750000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣␣0.000000
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣1.350000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣␣1.550000
H2␣␣␣␣␣␣␣␣␣␣␣␣␣1.350000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣-1.550000
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&SCF;␣Title="H2O␣TS␣optimization";␣KSDFT=B3LYP
␣&SLAPAF␣;␣ITER=20␣;␣TS
>>>␣EndDo
```

Notice that we have used the initial coordinates in Bohr, just to show that this is the default case. The optimal geometry for ground state water has C_{2v} symmetry. A transition state has been found with a linear HOH angle of 180^o. In many cases, as for example along the energy path for a chemical reaction, we may have a clue about the structure of the transition state, which typically represents an intermediate conformation between reactives and products. In that case we may try to orient the program leading it to the guessed solution. Keyword FindTS in `SLAPAF` has this goal. It has to be accompanied with a definition of constrained geometric definitions. `SLAPAF` will guide the optimization of the transition state towards a region in which the restriction is fulfilled. Once there, the restriction will be released and a free search of the transition state will be performed. This technique is frequently quite effective and makes it possible to find difficult transition states or reduce the number of required iterations. Here we show an example in which the initial geometry of water is clearly bent, and we impose the trial restriction that the angle for the transition state should be near 180^o. The final transition state will, however, be obtained without any type of geometrical restriction.

```
*DFT␣transition␣state␣optimization␣of␣the␣water␣molecule␣with␣geometrical␣restrictions
*File:␣DFT.transition_state_restricted.H2O
*
␣&Gateway
Title=␣H2O␣TS␣optimization
Basis␣set
O.ANO-S-MB
O␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.250000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣␣0.000000
End␣of␣basis
Basis␣set
H.ANO-S-MB
H1␣␣␣␣␣␣␣␣␣␣␣␣␣1.350000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣␣1.550000
H2␣␣␣␣␣␣␣␣␣␣␣␣␣1.350000␣␣␣␣␣␣␣␣0.000000␣␣␣␣␣␣␣-1.550000
End␣of␣basis
␣Constraints
␣␣␣a␣=␣Angle␣H1␣O␣H2
␣Value
␣␣␣a␣=␣180.0␣degree
␣End␣of␣Constraints
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&SCF;␣Title="H2O␣TS␣optimization";␣KSDFT=B3LYP
␣&SLAPAF␣;FindTS
>>>␣EndDo
```

The `CASPT2` geometry optimizations are somewhat different because `ALASKA` is not suited to compute `CASPT2` analytical gradients. Therefore the `ALASKA` program is automatically substituted by program `CASPT2_GRADIENT`, which will take care of performing numerical gradients. From the user pointview the only requirent is to place the `CASPT2` input after the `RASSCF` input. The CASSCF wave function has of course to be generated in each step before performing CASPT2. To compute a numerical gradient can be quite time consuming, although it is a task that can be nicely parallelized. In a double-sided gradient algorithm like here a total of 6N+1 CASPT2 calculations are performed each pass of the optimization, where N is the number of atoms.

```
*CASPT2␣minimum␣energy␣optimization␣for␣water
*File:␣CASPT2.minimum_optimization.H2O
*
␣&GATEWAY
Title=␣H2O␣minimum␣optimization
Basis␣set
O.ANO-S...2s1p.
```

```
O␣␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣␣0.504284␣Angstrom
H2␣␣␣␣␣␣␣␣0.758602␣␣0.000000␣-0.504284␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&RASSCF;␣Title="H2O␣restricted␣minimum";␣nActEl=8␣0␣0;␣Inactive=1;␣Ras2=6
␣&CASPT2;␣Frozen=1
␣&SLAPAF
>>>␣EndDo
```

The use of spatial symmetry makes the calculations more efficient, although they may again complicate the preparation of input files. We can repeat the previous `CASPT2` optimization by restricting the molecule to work in the C_{2v} point group, which, by the way, is the proper symmetry for water in the ground state. The `GATEWAY` program (as no symmetry has been specified) will identify and work with the highest available point group, C_{2v}. Here the molecule is placed with YZ as the molecular plane. By adding keyword SYMMETRY containing as elements of symmetry the YZ (symbol X) and YX (symbol Z), the poing group is totally defined and the molecule properly generated. From that point the calculations will be restricted to use symmetry restrictions. For instance, the molecular orbitals will be classified in the four elements of symmetry of the group, a_1, b_1, b_2, and a_2, and most of the programs will require to define the selection of the orbitals in the proper order. The order of the symmetry labels is determined by `SEWARD` and must be checked before proceeding, because from that point the elements of symmetry will be known by their order in `SEWARD`: a_1, b_1, b_2, and a_2, for instance, will be symmetries 1, 2, 3, and 4, respectively. `SCF` does not require to specify the class of orbitals and it can be used as a learning tool.

```
*CASPT2␣minimum␣energy␣optimization␣for␣water␣in␣C2v
*File:␣CASPT2.minimum_optimization_C2v.H2O
*
␣&GATEWAY
Title=␣H2O␣caspt2␣minimum␣optimization
Symmetry=␣X␣Z
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣␣␣0.000000␣␣0.758602␣␣0.504284␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
␣&RASSCF;␣Title="H2O␣caspt2␣minimum␣optimization";␣nActEl=8␣0␣0;␣Inactive=1␣0␣0␣0;␣Ras2=3␣1␣2␣0
␣&CASPT2;␣Frozen=1␣0␣0␣0
␣&SLAPAF␣&END
>>>␣EndDo
```

Thanks to symmetry restrictions the number of iterations within `CASPT2_GRADIENT` has been reduced to nine instead of nineteen, because many of the deformations are redundant within the C_{2v} symmetry. Also, symmetry considerations are important when defining geometrical restrictions (see sections 8.34.4 and 5.2).

4.1.3 Computing excited states

The calculation of electronic excited states is typically a multiconfigurational problem, and therefore it should preferably be treated with multiconfigurational methods such as CASSCF and CASPT2. We can start this section by computing the low-lying electronic states of the acrolein molecule (see figure above) at the CASSCF level and using a minimal basis set. The standard file with cartesian coordinates is:

```
␣8
Angstrom
␣O␣␣␣␣␣␣␣-1.808864␣␣␣-0.137998␣␣␣␣0.000000
␣C␣␣␣␣␣␣␣␣1.769114␣␣␣␣0.136549␣␣␣␣0.000000
␣C␣␣␣␣␣␣␣␣0.588145␣␣␣-0.434423␣␣␣␣0.000000
␣C␣␣␣␣␣␣␣-0.695203␣␣␣␣0.361447␣␣␣␣0.000000
␣H␣␣␣␣␣␣␣-0.548852␣␣␣␣1.455362␣␣␣␣0.000000
␣H␣␣␣␣␣␣␣␣0.477859␣␣␣-1.512556␣␣␣␣0.000000
␣H␣␣␣␣␣␣␣␣2.688665␣␣␣-0.434186␣␣␣␣0.000000
␣H␣␣␣␣␣␣␣␣1.880903␣␣␣␣1.213924␣␣␣␣0.000000
```

We shall carry out State-Averaged (SA) CASSCF calculations, in which one single set of molecular orbitals is used to compute all the states of a given spatial and spin symmetry. The obtained density matrix is the average for all states included, although each state will have its own set of optimized CI coefficients. Different weights can be considered for each of the states, but this should not be used except in very special cases by experts. It is better to let the CASPT2 method to handle that. The use of a SA-CASSCF procedure has an great advantage. For example, all states in a SA-CASSCF calculation are orthogonal to each other, which is not necessarily true for state specific calculations. Here, we shall include five states of singlet character the calculation. As no symmetry is invoked all the states belong by default to the first symmetry, including the ground state.

```
*CASSCF␣SA␣calculation␣on␣five␣singlet␣excited␣states␣in␣acrolein
*File:␣CASSCF.excited.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
coord␣=␣acrolein.xyz;␣basis␣=␣STO-3;␣group␣=␣c1
␣&SEWARD;␣&SCF
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
␣&GRID_IT
All;␣Ascii
```

We have used as active all the π and π^* orbitals, two bonding and two antibonding π orbitals with four electrons and in addition the oxygen lone pair (n). Keyword CiRoot informs the program that we want to compute a total of five states, the ground state and the lowest four excited states at the CASSCF level and that all of them should have the same weight in the average procedure. Once analyzed we find that the calculation has provided, in this order, the ground state, two $n \rightarrow \pi^*$ states, and two $\pi \rightarrow \pi^*$ states. It is convenient to add the `GRID_IT` input in order to be able to use the `GV` interface for the analysis of the orbitals and the occupations in the different electronic states. Such an analysis should always be made in order to understand the nature of the different excited states. In order to get a more detailed analysis of the nature of the obtained states it is also possible to obtain in a graphical way the charge density differences between to states, typically the difference between the ground and an excited state. The following example creates five different density files:

```
*CASSCF␣SA␣calculation␣on␣five␣singlet␣excited␣states␣in␣acrolein
*File:␣CASSCF.excited_grid.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
coord=␣acrolein.xyz;␣basis=␣STO-3G;␣group=␣c1
␣&SEWARD;␣&SCF
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
OutOrbital
Natural=␣5
␣&GRID_IT
FILEORB␣=␣$Project.RasOrb;␣NAME␣=␣$Project.grid
All
␣&GRID_IT
FILEORB␣=␣$Project.RasOrb2;␣NAME␣=␣$Project.grid2
All
␣&GRID_IT
FILEORB␣=␣$Project.RasOrb3;␣NAME␣=␣$Project.grid3
All
␣&GRID_IT
FILEORB␣=␣$Project.RasOrb4;␣NAME␣=␣$Project.grid4
All
␣&GRID_IT
FILEORB␣=␣$Project.RasOrb5;␣NAME=␣$Project.grid5
All
```

In `GRID_IT` input we have included all orbitals. It is, however, possible and in general recommended to restrict the calculation to certain sets of orbitals. How to do this is described in the input manual for `GRID_IT`. If we use now the `GV` program for instance with the command `molcas gv $Project.grid -a -1.0 $Project.grid2` the graph will display the differential charge transfer from the ground to the first excited state, in which the charge will leave the oxygen lone pair toward the π structure (-1.0 is the combination factor between both densities, that is, the difference).

CASSCF wave functions are typically good enough, but this is not the case for electronic energies, and the dynamic correlation effects have to be included, in particular here with the CASPT2 method. The proper input is prepared, again including `SEWARD` and `RASSCF` (unnecessary if they were computed previously), adding a `CASPT2` input with the keyword MultiState set to 5 1 2 3 4 5. The `CASPT2` will perform four consecutive single-state (SS) CASPT2 calculations using the SA-CASSCF roots computed by the `RASSCF` module. At the end, a multi-state CASPT2 calculation will be added in which the five SS-CASPT2 roots will be allowed to interact. The final MS-CASPT2 solutions, unlike the previous SS-CASPT2 states, will be orthogonal. The FROZen keyword is put here as a reminder. By default the program leaves the core orbitals frozen.

```
*CASPT2␣calculation␣on␣five␣singlet␣excited␣states␣in␣acrolein
*File:␣CASPT2.excited.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
coord␣=␣acrolein.xyz;␣basis␣=␣STO-3G;␣group=␣c1
␣&SEWARD;␣&SCF
```

```
␣&RASSCF
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
␣&GRID_IT
All;␣ASCII
␣&CASPT2
Title=␣Acrolein␣molecule
Multistate=␣5␣1␣2␣3␣4␣5
Frozen=␣4
```

Apart from energies and state properties it is quite often necessary to compute state interaction properties such as transition dipole moments, Einstein coefficients, and many other. This can be achieved with the `RASSI` module, a powerful program which can be used for many purposes (see section 8.31). We can start by simply computing the basic interaction properties

```
*RASSI␣calculation␣on␣five␣singlet␣excited␣states␣in␣acrolein
*File:␣RASSI.excited.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
coord␣=␣acrolein.xyz;␣basis␣=␣STO-3G;␣group␣=␣c1
␣&SEWARD;␣&SCF
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
␣&CASPT2
Title␣=␣caspt2
Frozen␣=␣4
MultiState=␣5␣1␣2␣3␣4␣5
>>LINK␣-FORCE␣$Project.JobMix␣JOB001
␣&RASSI
Nr␣of␣JobIph
␣1␣5
␣1␣2␣3␣4␣5
EJob
```

Oscillator strengths for the computed transitions and Einstein coefficients are compiled at the end of the `RASSI` output file. To obtain these values, however, energy differences have been used which are obtained from the previous CASSCF calculation. Those energies are not accurate because they do not include dynamic correlation energy and it is better to substitute them by properly computed values, such those at the CASPT2 level. This is achieved with the keyword EJob. More information is available in section 5.5.1.

Now a more complex case. We want to compute vertical singlet-triplet gaps from the singlet ground state of acrolein to different, up to five, triplet excited states. Also, interaction properties are requested. Considering that the spin multiplicity differs from the ground to the excited states, the spin Hamiltonian has to be added to our calculations and the `RASSI` program takes charge of that. It is required first, to add in the `SEWARD` input the keyword AMFI, which introduces the proper integrals required, and to the `RASSI` input the keyword SpinOrbit. Additionally, as we want to perform the calculation sequentially and `RASSI` will read from two different wave function calculations, we need to perform specific links to save the information. The link to the first `CASPT2` calculation will saved in file *`$Project.JobMix.S`* the data from the `CASPT2` result of the ground state, while the second link before the second `CASPT2` run will do the same for the triplet states. Later, we link these files as *`JOB001`* and

JOB002 to become input files for `RASSI`. In the `RASSI` input NrofJobIph will be set to two, meaning two *JobIph* or *JobMix* files, the first containing one root (the ground state) and the second five roots (the triplet states). Finally, we have added EJob, which will read the CASPT2 (or MS-CASPT2) energies from the *JobMix* files to be incorporated to the `RASSI` results. The magnitude of properties computed with spin-orbit coupling (SOC) depends strongly on the energy gap, and this has to be computed at the highest possible level, such as CASPT2.

```
*CASPT2/RASSI␣calculation␣on␣singlet-triplet␣gaps␣in␣acrolein
*File:␣CASPT2.S-T_gap.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
coord␣=␣acrolein.xyz;␣basis␣=␣STO-3G;␣group=␣c1
␣&SEWARD
AMFI
␣&SCF
␣&RASSCF
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣1␣1␣1
>>LINK␣-FORCE␣$Project.JobMix.S␣␣␣JOBMIX
␣&CASPT2
Title=␣acrolein
Frozen=␣4
MultiState=␣1␣1
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣3;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
>>LINK␣-FORCE␣$Project.JobMix.T␣␣␣JOBMIX
␣&CASPT2
Title=␣acrolein
Frozen=␣4
MultiState=␣5␣1␣2␣3␣4␣5
>>LINK␣-FORCE␣$Project.JobMix.S␣␣␣JOB001
>>LINK␣-FORCE␣$Project.JobMix.T␣␣␣JOB002
␣&RASSI
Nr␣of␣JobIph=␣2␣1␣5;␣1;␣1␣2␣3␣4␣5
Spin
EJob
```

As here with keyword AMFI, when using command Coord to build a `SEWARD` input and we want to introduce other keywords, it is enough if we place them after the line corresponding to Coord. Observe that the nature of the triplet states obtained is in sequence one $n\pi^*$, two $\pi\pi^*$, and two $n\pi^*$. The `RASSI` output is somewhat complex to analyze, but it makes tables summarizing oscillator strengths and Einstein coefficients, if those are the magnitudes of interest. Notice that a table is first done with the spin-free states, while the final table include the spin-orbit coupled eigenstates (in the CASPT2 energy order here), in which each former triplet state has three components.

In many cases working with symmetry will help us to perform calculations in quantum chemistry. As it is a more complex and delicate problem we direct the reader to the examples section in this manual. However, we include here two inputs that can help the beginners. They are based on trans-1,3-butadiene, a molecule with a C_{2h} ground state. If we run the next input, the `SEWARD` and `SCF` outputs will help us to understand how orbitals are classified by symmetry, whereas reading the `RASSCF` output the structure of the active space and states will be clarified.

```
*CASSCF␣SA␣calculation␣on␣1Ag␣excited␣states␣in␣tButadiene
*File:␣CASSCF.excited.tButadiene.1Ag
*
␣&SEWARD
Title=␣t-Butadiene␣molecule
Symmetry=␣Z␣XYZ
Basis␣set
C.STO-3G...
C1␣␣␣-3.2886930␣-1.1650250␣0.0000000␣␣Bohr
C2␣␣␣-0.7508076␣-1.1650250␣0.0000000␣␣Bohr
End␣of␣basis
Basis␣set
H.STO-3G...
H1␣␣␣-4.3067080␣␣0.6343050␣0.0000000␣␣Bohr
H2␣␣␣-4.3067080␣-2.9643550␣0.0000000␣␣Bohr
H3␣␣␣␣0.2672040␣-2.9643550␣0.0000000␣␣Bohr
End␣of␣basis
␣&SCF
␣&RASSCF
LumOrb
Title=␣tButadiene␣molecule␣(1Ag␣states);␣Symetry␣order␣(ag␣bg␣bu␣au)
Spin=␣1;␣Symmetry=␣1;␣Nactel=␣4␣0␣0;␣Inactive=␣7␣0␣6␣0;␣Ras2=␣0␣2␣0␣2
CiRoot=␣4␣4␣1
␣&GRID_IT
All;␣Ascii
```

Using the next input will give information about states of a different symmetry. Just run it as a simple exercise.

```
*CASSCF␣SA␣calculation␣on␣1Bu␣excited␣states␣in␣tButadiene
*File:␣CASSCF.excited.tButadiene.1Bu
*
␣&SEWARD
Title=␣t-Butadiene␣molecule
Symmetry=␣Z␣XYZ
Basis␣set
C.STO-3G...
C1␣␣␣-3.2886930␣-1.1650250␣0.0000000␣␣Bohr
C2␣␣␣-0.7508076␣-1.1650250␣0.0000000␣␣Bohr
End␣of␣basis
Basis␣set
H.STO-3G...
H1␣␣␣-4.3067080␣␣0.6343050␣0.0000000␣␣Bohr
H2␣␣␣-4.3067080␣-2.9643550␣0.0000000␣␣Bohr
H3␣␣␣␣0.2672040␣-2.9643550␣0.0000000␣␣Bohr
End␣of␣basis
␣&SCF
>LINK␣FORCE␣$Project.1Ag.JobIph␣␣JOBIPH
>LINK␣FORCE␣$Project.1Ag.RasOrb␣␣RASORB
␣&RASSCF
LumOrb
Title=␣tButadiene␣molecule␣(1Bu␣states);␣Symetry␣order␣(ag␣bg␣bu␣au)
Spin=␣1;␣Symmetry=␣1;␣Nactel=␣4␣0␣0;␣Inactive=␣7␣0␣6␣0
Ras2=␣0␣2␣0␣2
CiRoot=␣4␣4␣1
␣&GRID_IT
Name=␣$Project.1Ag.grid
All
>LINK␣FORCE␣$Project.1Bu.JobIph␣␣JOBIPH
>LINK␣FORCE␣$Project.1Bu.RasOrb␣␣RASORB
␣&RASSCF
LumOrb
Title=␣tButadiene␣molecule␣(1Bu␣states);␣Symetry␣order␣(ag␣bg␣bu␣au)
Spin=␣1;␣Symmetry=␣3;␣Nactel=␣4␣0␣0;␣Inactive=␣7␣0␣6␣0;␣Ras2=␣0␣2␣0␣2
CiRoot=␣2␣2␣1
```

```
␣&GRID_IT
Name=␣$Project.1Bu.grid
All;␣Ascii
>LINK␣FORCE␣$Project.1Ag.JobIph␣␣JOB001
>LINK␣FORCE␣$Project.1Bu.JobIph␣␣JOB002
␣&RASSI
NrofJobIph=␣2␣4␣2;␣1␣2␣3␣4;␣1␣2
```

Structure optimizations can be also performed at the CASSCF, RASSCF or CASPT2 levels. Here we shall optimize the second singlet state in the first (here the only) symmetry for acrolein at the SA-CASSCF level. It is strongly recommended to use the State-Average option and avoid single state CASSCF calculations for excited states. Those states are non-orthogonal with the ground state and are typically heavily contaminated. The usual set of input commands will be prepared, with few changes. In the `RASSCF` input two states will be simultaneously computed with equal weight (CiRoot 2 2 1), but, in order to get accurate gradients for a specific root (not an averaged one), we have to add Rlxroot and set it to two, which is, among the computed roots, that we want to optimize. The proper density matrix will be stored. The `MCLR` program optimizes, using a perturbative approach, the orbitals for the specific root (instead of using averaged orbitals), but the program is called automatically and no input is needed.

```
*CASSCF␣excited␣state␣optimization␣in␣acrolein
*File:␣CASSCF.excited_state_optimization.acrolein
*
␣&GATEWAY
Title=␣acrolein␣minimum␣optimization␣in␣excited␣state␣2
Basis␣set
O.STO-3G...2s1p.
O1␣␣␣␣␣␣␣1.608542␣␣␣␣␣␣-0.142162␣␣␣␣␣␣␣3.240198␣Angstrom
End␣of␣basis
Basis␣set
C.STO-3G...2s1p.
C1␣␣␣␣␣␣-0.207776␣␣␣␣␣␣␣0.181327␣␣␣␣␣␣-0.039908␣Angstrom
C2␣␣␣␣␣␣␣0.089162␣␣␣␣␣␣␣0.020199␣␣␣␣␣␣␣1.386933␣Angstrom
C3␣␣␣␣␣␣␣1.314188␣␣␣␣␣␣␣0.048017␣␣␣␣␣␣␣1.889302␣Angstrom
End␣of␣basis
Basis␣set
H.STO-3G...1s.
H1␣␣␣␣␣␣␣2.208371␣␣␣␣␣␣␣0.215888␣␣␣␣␣␣␣1.291927␣Angstrom
H2␣␣␣␣␣␣-0.746966␣␣␣␣␣␣-0.173522␣␣␣␣␣␣␣2.046958␣Angstrom
H3␣␣␣␣␣␣-1.234947␣␣␣␣␣␣␣0.213968␣␣␣␣␣␣-0.371097␣Angstrom
H4␣␣␣␣␣␣␣0.557285␣␣␣␣␣␣␣0.525450␣␣␣␣␣␣-0.720314␣Angstrom
>>>␣Do␣while
␣&SEWARD
>>>␣If␣(␣Iter␣=␣1␣)␣<<<
␣&SCF
Title=␣acrolein␣minimum␣optimization
>>>␣EndIf␣<<<
␣&RASSCF
LumOrb
Title=␣acrolein
Spin=␣1;␣nActEl=␣4␣0␣0;␣Inactive=␣13;␣Ras2=␣4
CiRoot=␣2␣2␣1
Rlxroot=␣2
␣&SLAPAF
>>>␣EndDo
```

In case of performing a `CASPT2` optimization for an excited state, still the SA-CASSCF approach can be used to generate the reference wave function, but keyword Rlxroot and the use of the `MCLR` program are not necessary, because `CASPT2` takes care of selecting the proper root.

A very useful tool recently included in *MOLCAS* is the possibility to compute minimum energy paths (MEP), representing steepest descendent minimum energy reaction paths which are built through a series of geometry optimizations, each requiring the minimization of the potential energy on a hyperspherical cross section of the PES centered on a given reference geometry and characterized by a predefined radius. One usually starts the calculation from a high energy reference geometry, which may correspond to the Franck-Condon (FC) structure on an excited-state PES or to a transition structure (TS). Once the first lower energy optimized structure is converged, this is taken as the new hypersphere center, and the procedure is iterated until the bottom of the energy surface is reached. Notice that in the TS case a pair of steepest descent paths, connecting the TS to the reactant and product structures (following the forward and reverse orientation of the direction defined by the transition vector) provides the minimum energy path (MEP) for the reaction. As mass-weighted coordinates are used by default, the MEP coordinate corresponds to the so-called Intrinsic Reaction Coordinates (IRC). We shall compute here the MEP from the FC structure of acrolein along the PES of the second root in energy at the CASSCF level. It is important to remember that the CASSCF order may not be accurate and the states may reverse orders at higher levels such as CASPT2.

```
*CASSCF␣excited␣state␣mep␣points␣in␣acrolein
*File:␣CASSCF.mep_excited_state.acrolein
*
␣&GATEWAY
Title␣=␣acrolein␣mep␣calculation␣root␣2
Basis␣set
O.STO-3G...2s1p.
␣O1␣␣␣␣1.367073␣␣␣␣␣0.000000␣␣␣␣␣3.083333␣Angstrom
End␣of␣basis
Basis␣set
C.STO-3G...2s1p.
␣C1␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣0.000000␣Angstrom
␣C2␣␣␣␣0.000000␣␣␣␣␣0.000000␣␣␣␣␣1.350000␣Angstrom
␣C3␣␣␣␣1.367073␣␣␣␣␣0.000000␣␣␣␣␣1.833333␣Angstrom
End␣of␣basis
Basis␣set
H.STO-3G...1s.
␣H1␣␣␣␣2.051552␣␣␣␣␣0.000000␣␣␣␣␣0.986333␣Angstrom
␣H2␣␣␣-0.684479␣␣␣␣␣0.000000␣␣␣␣␣2.197000␣Angstrom
␣H3␣␣␣-1.026719␣␣␣␣␣0.000000␣␣␣␣-0.363000␣Angstrom
␣H4␣␣␣␣0.513360␣␣␣␣␣0.889165␣␣␣␣-0.363000␣Angstrom
End␣of␣basis
>>>␣Set␣maxiter␣100
>>>␣Do␣while
␣&SEWARD
>>>␣If␣(␣Iter␣=␣1␣)␣<<<
␣&SCF␣;␣Title="Acrolein␣mep␣calculation␣root␣2"
>>>␣EndIf␣<<<
␣&RASSCF␣;␣Title="acrolein␣mep␣calculation␣root␣2";␣Spin=1
␣␣␣␣␣␣␣␣␣␣␣nActEl=4␣0␣0;␣Inactive=13;␣Ras2=4;␣CiRoot=2␣2␣1;␣Rlxroot=2
␣&SLAPAF
␣␣␣␣␣Constraint
␣␣␣␣␣␣␣␣a␣=␣Sphere
␣␣␣␣␣Value
␣␣␣␣␣␣␣␣a␣=␣0.1
␣␣␣␣␣End␣of␣Constraints
␣␣␣␣␣MEP-search
>>>␣EndDo
```

As observed, to prepare the input for the MEP is simple, just to add the keyword MEP-

SEARCH and specify a hypersphere constraint to the `SLAPAF` input, and the remaining structure equals that of a geometry optimization. The calculations are time consuming, because each point of the MEP (four plus the initial one obtained here) is computed through a specific optimization. A file named *`$Project.mep.molden`* (read by `MOLDEN`) will be generated in $WorkDir containing only those points belonging to the MEP.

We shall now show how to perform geometry optimizations under nongeometrical restrictions, in particular, how to compute hypersurface crossings, which are key structures in the photophysics of molecules. We shall get those points as minimum energy crossing points in which the energy of the highest of the two states considered is minimized under the restriction that the energy difference with the lowest state should equal certain value (typically zero). Such point can be named a minimum energy crossing point (MECP). If a further restriction is imposed, like the distance to a specific geometry, and several MECP as computed at varying distances, it is possible to obtain a crossing seam of points where the energy between the two states is degenerated. Those degeneracy points are funnels with the highest probability for the energy to hop between the surfaces in internal conversion or intersystem crossing photophysical processes. There are different possibilities. A crossing between states of the same spin multiplicity and spatial symmetry is named a conical intersection. Elements like the nonadiabatic coupling terms are required to obtain them strictly, and they are not computed presently by *MOLCAS*. If the crossing occurs between states of the same spin multiplicity and different spatial symmetry or between states of different spin multiplicity, the crossing is an hyperplane and its only requirement is the energetic degeneracy and the proper energy minimization.

Here we include an example with the crossing between the lowest singlet (ground) and triplet states of acrolein. Notice that two different states are computed, first by using `RASSCF` to get the wave function and then `ALASKA` to get the gradients of the energy. Nothing new on that, just the information needed in any geometry optimizations. The `SLAPAF` input requires to add as constraint an energy difference between both states equal to zero. A specific instruction is required after calculating the first state. We have to copy the communication file *`RUNFILE`* (at that point contains the information about the first state) to *`RUNFILE2`* to provide later `SLAPAF` with proper information about both states:

```
*CASSCF␣singlet-triplet␣crossing␣in␣acrolein
*File:␣CASSCF.S-T_crossing.acrolein
*
␣&GATEWAY
Title=␣Acrolein␣molecule
Basis␣set
O.sto-3g....
␣O1␣␣␣␣␣␣␣␣␣␣␣␣␣1.5686705444␣␣␣␣␣␣␣-0.1354553340␣␣␣␣␣␣␣␣3.1977912036␣␣Angstrom
End␣of␣basis
Basis␣set
C.sto-3g....
␣C1␣␣␣␣␣␣␣␣␣␣␣␣-0.1641585340␣␣␣␣␣␣␣␣0.2420235062␣␣␣␣␣␣␣-0.0459895824␣␣Angstrom
␣C2␣␣␣␣␣␣␣␣␣␣␣␣␣0.1137722023␣␣␣␣␣␣␣-0.1389623714␣␣␣␣␣␣␣␣1.3481527296␣␣Angstrom
␣C3␣␣␣␣␣␣␣␣␣␣␣␣␣1.3218729238␣␣␣␣␣␣␣␣0.1965728073␣␣␣␣␣␣␣␣1.9959513294␣␣Angstrom
End␣of␣basis
Basis␣set
H.sto-3g....
␣H1␣␣␣␣␣␣␣␣␣␣␣␣␣2.0526602523␣␣␣␣␣␣␣␣0.7568282320␣␣␣␣␣␣␣␣1.4351034056␣␣Angstrom
␣H2␣␣␣␣␣␣␣␣␣␣␣␣-0.6138178851␣␣␣␣␣␣␣-0.6941171027␣␣␣␣␣␣␣␣1.9113821810␣␣Angstrom
␣H3␣␣␣␣␣␣␣␣␣␣␣␣-0.8171509745␣␣␣␣␣␣␣␣1.0643342316␣␣␣␣␣␣␣-0.2648232855␣␣Angstrom
␣H4␣␣␣␣␣␣␣␣␣␣␣␣␣0.1260134708␣␣␣␣␣␣␣-0.4020589690␣␣␣␣␣␣␣-0.8535699812␣␣Angstrom
End␣of␣basis
>>>␣Do␣while
```

```
␣&SEWARD
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF
Title
␣Acrolein␣S-T_crossing
>>>␣ENDIF␣<<<
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣4␣0␣0;␣Inactive=␣13;␣Ras2=␣4
CiRoot=␣1␣1;␣1
␣&ALASKA
>COPY␣$WorkDir/$Project.RunFile␣$WorkDir/RUNFILE2
␣&RASSCF
LumOrb
Title=␣Acrolein␣molecule
Spin=␣3;␣Nactel=␣4␣0␣0;␣Inactive=␣13;␣Ras2=␣4
CiRoot=␣1␣1;␣1
␣&ALASKA
␣&SLAPAF
Constraints
␣␣␣a␣=␣Ediff
␣␣Value
␣␣␣a␣=␣0.000
End␣of␣Constraints
>>>␣EndDo
```

Solvent effects can be also applied to excited states, but first the reaction field in the ground (initial) state has to be computed. This is because solvation in electronic excited states is a non equilibrium situation in with the electronic polarization effects (fast part of the reaction field) have to treated apart (they supposedly change during the excitation process) from the orientational (slow part) effects. The slow fraction of the reaction field is maintained from the initial state and therefore a previous calculation is required. From the practical point of view the input is simple as illustrated in the next example. First, the proper reaction-field input is included in `SEWARD`, then a `RASSCF` and `CASPT2` run of the ground state, with keyword RFPErt in `CASPT2`, and after that another SA-CASSCF calculation of five roots to get the wave function of the excited states. Keyword NONEequilibrium tells the program to extract the slow part of the reaction field from the previous calculation of the ground state (specifically from the *JOBOLD* file, which may be stored for other calculations) while the fast part is freshly computed. Also, as it is a SA-CASSCF calculation (if not, this is not required) keyword RFRoot is introduced to specify for which of the computed roots the reaction field is generated. We have selected here the fifth root because it has a very large dipole moment, which is also very different from the ground state dipole moment. If you compare the excitation energy obtained for the isolated and the solvated system, a the large red shift is obtained in the later.

```
*CASPT2␣excited␣state␣in␣water␣for␣acrolein
*File:␣CASPT2.excited_solvent.acrolein
*
␣&SEWARD
Title=␣Acrolein␣molecule
coord␣=␣acrolein.xyz;␣basis␣=␣STO-3G;␣group=␣c1
RF-input
␣␣␣PCM-model;␣solvent=␣water
End␣of␣RF-input
␣&RASSCF
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣1␣1␣1
```

```
␣&CASPT2
Title=␣Acrolein␣molecule
Multistate=␣1␣1
RFPert
␣&RASSCF
Title=␣Acrolein␣molecule
Spin=␣1;␣Nactel=␣6␣0␣0;␣Inactive=␣12;␣Ras2=␣5
CiRoot=␣5␣5␣1
RFRoot=␣5
NONEquilibrium
␣&CASPT2
Title=␣Acrolein␣molecule
Multistate=␣1␣5
RFPert
```

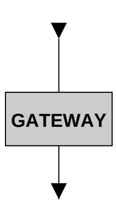
A number of simple examples as how to proceed with the most frequent quantum chemical problems computed with MOLCAShave been given above. Certainly there are many more possibilities in MOLCAS 7.4 such as calculation of 3D band systems in solids at a semiempirical level, obaining valence-bond structures, the use of QM/MM methods in combination with a external MM code, the introduction of external homogeneous or non homogeneous perturbations, generation of atomic basis sets, application of different localization schemes, analysis of first order polarizabilities, calculation of vibrational intensities, analysis, generation, and fitting of potentials, computation of vibro-rotational spectra for diatomic molecules, introduction of relativistic effects, etc. All those aspects are explained in the manual and are much more specific. Next section 4.2 details the basic structure of the inputs, program by program, while easy examples can also be found. Later, another chapter includes a number of extremely detailed examples with more elaborated quantum chemical examples, in which also scientific comments are included. Examples include calculations on high symmetry molecules, geometry optimizations and Hessians, computing reaction paths, high quality wave functions, excited states, solvent models, and computation of relativistic effects.

4.2 Program-based tutorials

The MOLCAS 7.4 suite of quantum chemical programs is modular in design. The desired calculation is achieved by executing a list of MOLCAS program modules in succession, occasionally manipulating the program information files. If the information files from a previous calculation are saved then a subsequent calculation need not recompute them. This is dependent on the correct information being preserved in the information files for the subsequent calculations. Each module has keywords to specify the functions to be carried out and many modules are reliant on the specification of keywords in previous modules.

The following sections describe the use of the MOLCAS modules and their interelationships. Each module is introduced in the approximate order of a calculation. The flow charts in each section show diagrammatically the dependencies of the module. A complete flowchart for the 7.4 suite of codes appears in Section 8.10.

4.2.1 GATEWAY — A Molecule Handling Program



The program `GATEWAY` handles the basic molecular parameters in the calculation. It generates data that are used in all subsequent calculations. These data are stored on the *`RUNFILE`*. `GATEWAY` is the first program you run (if you have not already generated the $WorkDir directory and the ***RUNFILE*** in an earlier calculation).

We commence this tutorial by setting up the basic data for a water molecule. For a more general description of the input options for `GATEWAY` we refer to the Users Guide. The input for water is given in Figure 4.1. The first line of the input is the program identifier &GATEWAY. Then follows the keyword used is TITLE. The first line following the keyword is printed in the `GATEWAY` section of the calculation output. Further, the first title line is saved in the integral file and appears in any subsequent programs. By convention we shall indent all none keyword entries.

The GROUP keyword is followed by the generators for the C_{2v} point group. The specification of the C_{2v} point group given in Table 4.1 is not unique, however. In this tutorial, the generators have been input in an order that reproduces the ordering in the character tables. A complete list of symmetry generator input syntax is given in Table 4.1. The symmetry groups available are listed with the symmetry generators defining the group. The *MOLCAS* keywords required to specify the symmetry groups are also listed. The last column contains the symmetry elements generated by the symmetry generators.

Figure 4.1: Sample input requested by the GATEWAY module for water in C_{2v} symmetry.

```
␣&GATEWAY
Title
␣Water␣in␣C2v␣symmetry␣-␣A␣Tutorial
Coord␣=␣water.xyz
Group␣=␣␣XY␣Y
Basis␣Set␣=␣O.ANO-S-MB,H.ANO-S-MB
```

Table 4.1: Symmetries available in MOLCAS including generators, MOLCAS keywords and symmetry elements.

Group	Generators			*MOLCAS*			Elements							
	g_1	g_2	g_3	g_1	g_2	g_3	E	g_1	g_2	g_1g_2	g_3	g_1g_3	g_2g_3	$g_1g_2g_3$
C_1							E							
C_2	C_2			XY			E	C_2						
C_s	σ			X			E	σ						
C_i	i			XYZ			E	i						
C_{2v}	C_2	σ_v		XY	Y		E	C_2	σ_v	σ_v'				
C_{2h}	C_2	i		XY	XYZ		E	C_2	i	σ_h				
D_2	C_2^z	C_2^y		XY	XZ		E	C_2^z	C_2^y	C_2^x				
D_{2h}	C_2^z	C_2^y	i	XY	XZ	XYZ	E	C_2^z	C_2^y	C_2^x	i	σ^{xy}	σ^{xz}	σ^{yz}

To reduce the input, the unity operator E is always assumed. The twofold rotation about the z-axis, $C_2(z)$, and the reflection in the xz-plane, $\sigma_v(xz)$, are input as XY and Y respectively. The *MOLCAS* input can be viewed as symmetry operators that operate on the Cartesian elements specified. For example, the reflection in the xz-plane is specified by the input keyword Y which is the Cartesian element operated upon by the reflection.

The input used in Figure 4.1 produces the character table in the `GATEWAY` section of the output shown in Figure 4.2. Note that $\sigma_v(yz)$ was produced from the other two generators. The last column contains the basis functions of each irreducible symmetry representation.

Figure 4.2: Character Table for C_{2v} from MOLCAS output.

```
           E   C2(z) s(xz) s(yz)
a1         1     1     1     1  z
b1         1    -1     1    -1  x, xz, Ry
a2         1     1    -1    -1  xy, Rz, I
b2         1    -1    -1     1  y, yz, Rx
```

The totally symmetric a_1 irreducible representation has the z basis function listed which is unchanged by any of the symmetry operations.

The geometry of the molecule is defined using the keyword COORD. On the next line we give the name of the xyz file that defines the geometrical parameters of the molecule (in this case `water.xyz`) We show this file for the water molecule in Figure 4.3. The first line contains the number of atoms. The second line is used to indicate the unit used: Ångström or atomic units. The default is to useÅngström. Then follows the name of each atom and its coordinates.

The basis sets are defined after the keyword BASIs SETS. The oxygen and hydrogen basis set we have chosen are the small Atomic Natural Orbitals (ANO) sets. There are three contractions of the basis included in the input (and commented out according to the desired calculation) which may be considered, the minimal, double zeta with polarization and triple zeta with polarization.

Figure 4.3: The geometry of the water molecule

```
 3

O         .000000         .000000        .000000
H        0.700000         .000000       0.700000
H       -0.700000         .000000       0.700000
```

`GATEWAY` Output

The `GATEWAY` output contains the symmetry character table, basis set information and input atomic centers. The basis set information lists the exponents and contraction coefficients as well as the type of Gaussian functions (Cartesian, spherical or contaminated) used.

The inter-nuclear distances and valence bond angles (including dihedral angles) are displayed after the basis set information. There is a keyword, RTRN, which is used to increase the threshold for printing of bond lengths, bond angles and dihedral angles from the default of 3.5 au. Inertia and rigid-rotor analysis is also included in the output along with the timing information.

A section of the output that is useful for determining the input to the *MOLCAS* module `SCF` is the symmetry adapted basis functions which appears near the end of the `GATEWAY` portion of the output. This is covered in more detail in the `SCF` tutorial.

The most important file produced by the `GATEWAY` module is the `RUNFILE` which in our case is linked to `water.RunFile`. This is the general *MOLCAS* communications file for transferring data between the various *MOLCAS* program modules. Many of the program modules add data to the ***RUNFILE*** which can be used in still other modules. A new ***RUNFILE*** is produced every time `GATEWAY` is run. It should finally be mentioned that for backwards compatibility one can run `MOLCAS` without invoking `GATEWAY`. The corresponding input and output will then be handled by the program `SEWARD`.

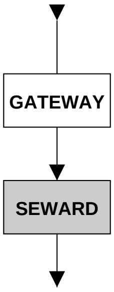
BSSE

`GATEWAY` can operates with several coordinate files, which is convenient for computing BSSE correction. BSSE followed by a number marks a XYZ file which should be treated as dummy atoms. The following example demonstrates this feature:

```
&GATEWAY
coord␣=␣ethanol.xyz
coord␣=␣water.xyz
bsse␣␣=␣1
basis␣=␣ANO-S-MB
NOMOVE
&SEWARD;␣&SCF
&GRID_IT
NAME␣=␣water
***************
&GATEWAY
coord␣=␣ethanol.xyz
coord␣=␣water.xyz
bsse␣␣=␣2
basis␣=␣ANO-S-MB
NOMOVE
&SEWARD;␣&SCF
&GRID_IT
NAME␣=␣ethanol
**************
&GATEWAY
coord␣=␣ethanol.xyz
coord␣=␣water.xyz
basis␣=␣ANO-S-MB
NOMOVE
&SEWARD;␣&SCF
&GRID_IT
NAME␣=␣akvavit
```

Note, that NOMOVE keyword prevents centering of the molecule, so the computed grids are identical. An alternative way to compute density difference is to modify coordinates, and change an element label to X.

4.2.2 SEWARD — An Integral Generation Program



An *ab initio* calculation always requires integrals. In the *MOLCAS* suite of programs, this function is supplied by the `SEWARD` module. `SEWARD` computes the one- and two-electron integrals for the molecule and basis set specified in the input to the program programGATEWAY, which should be run before `SEWARD`. `SEWARD` can also be used to perform some property expectation calculations on the isolated molecule. The module is also used as an input parser for the reaction field and numerical quadrature parameters.

We commence our tutorial by calculating the integrals for a water molecule. The input is given in Figure 4.4. Each MOLCAS module identifies input from a file by the name of the module. In the case of `SEWARD`, the program starts with the label &SEWARD, which is the first statement in the file shown below.

In normal cases no input is required for `SEWARD`, so the following input is optional. The first keyword used is TITLE. Only the first line of the title is printed in the output. The first title line is also saved in the integral file and appears in any subsequent programs that use the integrals calculated by `SEWARD`.

Figure 4.4: Sample input requesting the SEWARD module to calculate the integrals for water in C_{2v} symmetry.

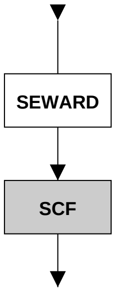
```
␣&SEWARD
Title
␣Water␣-␣A␣Tutorial
␣The␣integrals␣of␣water␣are␣calculated␣using␣C2v␣symmetry
```

In more complicated cases more input may be needed, to specify certain types of integrals, that use of Cholesky decomposition techniques, etc. We refer to the specific sections of the Users-Guide for more information. The output from a `SEWARD` calculation is small and contains in principle only a list of the different types of integrals that are computed.

The integrals produced by the `SEWARD` module are stored in two files in the working directory. They are ascribed the `FORTRAN` names *`ONEINT`* and *`ORDINT`* which are automatically symbolically linked by the MOLCAS script to the file names $Project*`.OneInt`* and $Project*`.OrdInt`*, respectively or more specifically, in our case, *`water.OneInt`* and *`water.OrdInt`*, respectively. The default name for each symbolical name is contained in the corresponding program files of the directory $MOLCAS/shell. The *`ONEINT`* file contains the one-electron integrals. The *`ORDINT`* contains the ordered and packed two-electron integrals. Both files are used by later MOLCAS program modules.

4.2.3 SCF — A Self-Consistent Field program and Kohn Sham DFT

The simplest *ab initio* calculations possible use the Hartree-Fock (HF) Self-Consistent Field (SCF) method with the program name `SCF` in the MOLCAS suite. It is possible to calculate the HF energy once we have calculated the integrals using the `SEWARD` module, although MOLCAS can perform a direct SCF calculation in which the two-electron integrals are not stored on disk. The MOLCAS implementation performs a closed-shell (all electrons are paired in orbitals) and open-shell (Unrestricted Hartree-Fock) calculation. It is not possible to perform an Restricted Open-shell Hartree-Fock (ROHF) calculation with the `SCF`. This is instead done using the program `RASSCF`. The `SCF` program can also be used to perform calculations using Kohn Sham Density Functional Theory (DFT).



The `SCF` input for a Hartree-Fock calculation of a water molecule is given in figure 4.5 which continues our calculations on the water molecule.

There are no compulsory keywords following the program name, &SCF. If no input is given the program will compute the SCF energy for a neutral molecule with the orbital occupations giving the lowest energ. Here, we have used the following input: the first is TITLE. As with the `SEWARD` program, the first line following the keyword is printed in the output.

The next keyword for `SCF` is OCCUPIED which specifies the number of occupied orbitals in each symmetry grouping listed in the `SEWARD` output and given in Figure 4.6. The basis label and type give an impression of the possible molecular orbitals that will be obtained in the SCF calculation. For example, the first basis function in the a_1 irreducible representation is an s type on the oxygen indicating the oxygen 1s orbital. Note, also, that the fourth basis function is centered on the hydrogens, has an s type and is symmetric on both hydrogens as indicated by both hydrogens having a phase of 1, unlike the sixth basis function which has a phase of 1 on center 2 (input H1) and -1 on center 3 (generated H1).

Figure 4.5: Sample input requesting the SCF module to calculate the ground Hartree-Fock energy for a neutral water molecule in C_{2v} symmetry.

```
␣&SCF␣&END
Title
␣Water␣-␣A␣Tutorial
␣The␣SCF␣energy␣of␣water␣is␣calculated␣using␣C2v␣symmetry
Occupied
␣3␣1␣0␣1
End␣of␣Input
```

Figure 4.6: Symmetry adapted Basis Functions from a SEWARD output.

```
          Irreducible representation : a1
          Basis function(s) of irrep: z

Basis Label        Type   Center Phase
  1   O1           1s0       1     1
  2   O1           1s0       1     1
  3   O1           2p0       1     1
  4   H1           1s0       2     1      3     1

          Irreducible representation : b1
          Basis function(s) of irrep: x, xz, Ry

Basis Label        Type   Center Phase
  5   O1           2p1+      1     1
  6   H1           1s0       2     1      3    -1

          Irreducible representation : b2
          Basis function(s) of irrep: y, yz, Rx

Basis Label        Type   Center Phase
  7   O1           2p1-      1     1
```

We have ten electrons to ascribe to five orbitals to describe a neutral water molecule in the ground state. Several techniques exist for correct allocation of electrons. As a test of the electron allocation, the energy obtained should be the same with and without symmetry. Water is a simple case, more so when using the minimal basis set. In this case, the third irreducible representation is not listed in the `SEWARD` output as there are no basis functions in that representation. That is why the third number after the OCCUPIED keyword is zero. As an alternative you can use the keyword CHARGE with parameters 0 and 1 to indicate a neutral molecule and optimization procedure 1 that searches for the optimal occupation.

To do a UHF calculation, the keyword UHF must be specified. In this case after a keyword OCCNUMBERS, you have to specify the default occupation numbers for alpha and beta orbitals. It is possible to use UHF together with keyword CHARGE or AUFBAU, in this case you have to specify a keyword ZSPIN, followed in the next line by the difference between alpha and beta electrons.

If you want to do an UHF calculation for a closed shell system, for example, diatomic molecule with large interatomic distance, you have to specify keyword SCRAMBLE.

To do the Density Functional Theory calculations, keyword KSDFT followed in the next line by the name of the available functional as listed in the input section is compulsory. Presently following Functional Keywords are available: BLYP, B3LYP, B3LYP5, HFB, HFS, LDA, LDA5, LSDA, LSDA5, SVWN, SVWN5, TLYP, XPBE, MO6, MO6/HF, MO6/2X, MO6/L. The description of functional keywords and the functionals is defined in the section DFT Calculations 8.33.2 The input for KSDFT is given as,

```
KSDFT
B3LYP5
```

In the above example B3LYP5 functional will be used in KSDFT calculations.

Running `SCF`

Performing the Hartree-Fock calculation introduces some important aspects of the transfer of data between the *MOLCAS* program modules. The `SCF` module uses the integral files computed by `SEWARD`. It produces a orbital file with the symbolic name *`SCFORB`* which contains all the MO information. This is then available for use in subsequent *MOLCAS* modules. The `SCF` module also adds information to the *`RUNFILE`*. Recall that the `SEWARD` module produces two integral files symbolically linked to *`ONEINT`* and *`ORDINT`* and actually called, in our case, *`water.OneInt`* and *`water.OrdInt`*, respectively. Because the two integral files are present in the working directory when the `SCF` module is performed, *MOLCAS* automatically links them to the symbolic names.

If the integral files were not deleted in a previous calculation the `SEWARD` calculation need not be repeated. Furthermore, integral files need not be in the working directory if they are linked by the user to their respective symbolic names. Integral files, however, are often very large making it desirable to remove them after the calculation is complete. The linking of files to their symbolic names is useful in other case, such as input orbitals.

If nothing else is stated, the `SCF` program will use the guess orbitals produced by `SEWARD` as input orbitals with the internal name *`GUESSORB`*. If one wants to use any other input orbitals for the `SCF` program the option LUMORB must be used. The corresponding file should be copied to the internal file *`INPORB`*. This could for example be an orbital file generated by an earlier SCF calculation, *`$Project.ScfOrb`*. Rename it to *`$Project.ScfOld`* and link this name to *`INPORB`*.

`SCF` Output

The `SCF` output includes the title from the input as well as the title from the `SEWARD` input because we used the integrals generated by `SEWARD`. The output also contains the cartesian coordinates of the molecule and orbital specifications including the number of frozen, occupied and virtual (secondary) orbitals in each symmetry. This is followed by details regarding the `SCF` algorithm including convergence criteria and iteration limits. The energy convergence information includes the one-electron, two-electron, and total energies for each iteration. This is followed by the final results including the final energy and molecular orbitals for each symmetry.

The Density Functional Theory Program gives in addition to the above, details of grids used, convergence criteria, and name of the functional used. This is followed by integrated DFT energy which is the functional contribution to the total energy and the total energy including the correlation. This is followed results including the Kohn Sham orbitals for each symmetry.

The molecular orbital (MO) information lists the orbital energy, the electron occupation and the coefficients of the basis functions contributing to that MO. For a minimal basis set, the basis functions correspond directly to the atomic orbitals. Using larger basis sets means that a combination of the basis functions will be used for each atomic orbital and more so for the MOs. The MOs from the first symmetry species are given in Figure 4.7. The first MO has an energy of -20.5611 hartree and an occupation of 2.0. The major contribution is from the first basis function label '`O1 1s0`' meaning an *s* type function centered on the oxygen atom. The orbital energy and the coefficient indicates that it is the MO based largely on the oxygen 1*s* atomic orbital.

Figure 4.7: Molecular orbitals from the first symmetry species of a calculation of water using C_{2v} symmetry and a minimal basis set.

```
     ORBITAL        1         2         3         4
     EneRGY     -20.5611   -1.3467    -.5957     .0000
     Occc. NO.    2.0000    2.0000    2.0000     .0000

   1 O1   1s0     1.0000    -.0131    -.0264    -.0797
   2 O1   1s0      .0011     .8608    -.4646    -.7760
   3 O1   2p0      .0017     .1392     .7809    -.7749
   4 H1   1s0     -.0009     .2330     .4849    1.5386
```

The second MO has a major contribution from the second oxygen `1s` basis function indicating a mostly oxygen 2*s* construction. Note that it is the absolute value of the coefficient that determines it importance. The sign is important for determining the orthogonality of its orbitals and whether the atomic orbitals contributions with overlap constructively (bonding) or destructively (anti-bonding). The former occurs in this MO as indicated by the positive sign on the oxygen 2*s* and the hydrogen 1*s* orbitals, showing a bonding interaction between them. The latter occurs in the third MO, where the relative sign is reversed.

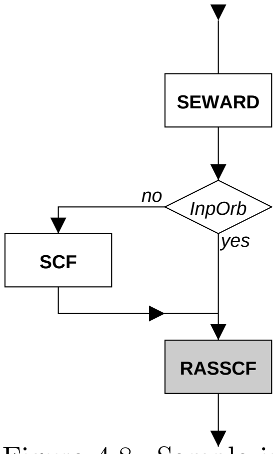
The third MO has an energy of -0.5957 hartree and major contributions from the second oxygen `1s` basis function, the oxygen `2p0` basis function and the hydrogen `1s` basis functions which are symmetrically situated on each hydrogen (see Figure 4.6). The mixing of the oxygen `2s` and `2p0` basis functions leads to a hybrid orbital that points away from the two hydrogens, to which it is weakly antibonding.

A similar analysis of the fourth orbital reveals that it is the strongly anti-bonding orbital partner to the third MO. The oxygen `2p0` basis function is negative which reverses the overlap characteristics.

The molecular orbital information is followed by a Mulliken charge analysis by input center and basis function. This provides a measure of the electronic charge of each atomic center.

Towards the end of the `SCF` section of the *MOLCAS* output various properties of the molecule are displayed. By default the first (dipole) and second cartesian moments and the quadrupoles are displayed. The inclusion of the FLDG keyword (with zero (0) on the next line) with cause the electric field gradients at each atomic center to be calculated and displayed. There are several other properties that can be calculated in this fashion using the variational *MOLCAS* programs – `SCF` and `RASSCF` when producing a CASSCF wave function.

4.2.4 RASSCF — A Multi Configurational Self-Consistent Field Program



One of the central codes in *MOLCAS* is the `RASSCF` program, which performes multiconfigurational SCF calculations. Both Complete Active Space (CASSCF) and Restricted Active Space (RASSCF) SCF calculations can be performed with the `RASSCF` program module [12]. An open shell Hartree-Fock calculation is not possible with the `SCF` but it can be performed using the `RASSCF` module. An input listing for a CASSCF calculation of water appears in Figure 4.8. `RASSCF` requires orbital information of the system which can be obtained in two ways. The LUMOrb inidcates that the orbitals should be taken from a user defined orbital file, which is copied to the internal file INPORB. If this keyword is not given, the program will look for orbitals on the runfile in the preference order: *`RASORB`*, *`SCFORB`* and *`GUESSORB`*

Figure 4.8: Sample input requesting the RASSCF module to calculate the eight-electrons-in-six-orbitals CASSCF energy of the second excited triplet state in the second symmetry group of a water molecule in C_{2v} symmetry.

```
␣&RASSCF␣&END
Title
␣The␣CASSCF␣energy␣of␣water␣is␣calculated␣using␣C2v␣symmetry.␣2␣3B2␣state.
Inactive
␣1␣0␣0␣0
Ras2
␣3␣2␣0␣1
nActEl
␣8␣␣0␣␣0
Symmetry
␣2
Spin
␣3
CIRoot
␣1␣2
␣2
LevShift
␣1.0
LumOrb
End␣of␣Input
```

The TITLe performs the same function as in the previous *MOLCAS*

Table 4.2: Examples of types of wave functions obtainable using the RAS1 and RAS3 spaces in the RASSCF module.

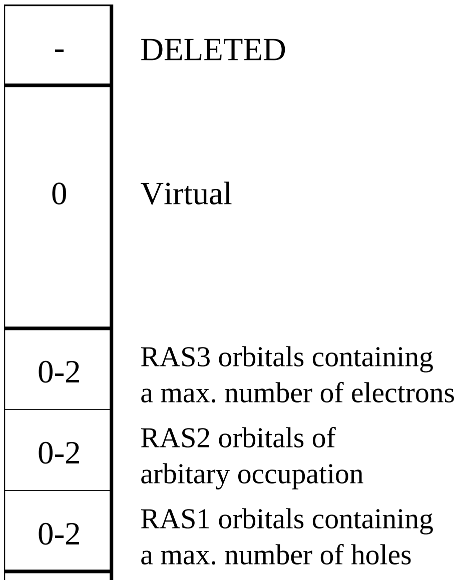
Description	Number of holes in RAS1 orbitals	RAS2 orbitals	Number of electrons in RAS3 orbitals
SD-CI	2	0	2
SDT-CI	3	0	3
SDTQ-CI	4	0	4
Multi Reference SD-CI	2	n	2
Multi Reference SD(T)-CI	3	n	2

modules. The keyword INACtive specifies the number of doubly occupied orbitals in each symmetry that will not be included in the electron excitations and thus remain doubly occupied throughout the calculation. A diagram of the complete orbital space available in the `RASSCF` module is given in Figure 4.9.

In our calculation, we have placed the oxygen 1s orbital in the inactive space using the INACtive keyword. The keyword FROZen can be used, for example, on heavy atoms to reduce the Basis Set Superposition Error (BSSE). The corresponding orbitals will then not be optimized. The RAS2 keyword specifies the number of orbitals in each symmetry to be included in the electron excitations with all possible occupations allowable. Because the RAS1 and RAS3 spaces are zero (not specified in the input in Figure 4.8) the `RASSCF` calculation will produce a CASSCF wave function. The RAS2 space is chosen to use all the orbitals available in each symmetry (expect the oxygen 1s orbital). The keyword NACTel specifies the number of active electrons (8), maximum number of holes in the Ras1 space (0) and the maximum number of electrons in the Ras3 space (0). Using the keywords RAS1 and/or RAS3 to specify orbitals and specifying none zero numbers of holes/electrons will produce a RASSCF wave function.We are, therefore, performing an 8in6 CASSCF calculation of water.

There are a number of wave function types that can be performed by manipulating the RAS1 and RAS3 spaces. Table 4.2 lists a number of types obtainable. The first three are Configuration Interaction (CI) wave functions of increasing magnitude culminating with a Single, Double, Triples and Quadruples (SDTQ) CI. These can become multi reference if the number of RAS2 orbitals is non-zero. The last type provides some inclusion of the triples excitation by allowing three holes in the RAS1 orbitals but save computation cost by only allowing double excitations in the RAS3 orbitals.

The symmetry of the wave function is specified using the SYMMetry keyword. It specifies the number of the symmetry subgroup in the calculation. We have chosen the second symmetry species, b$_2$, for this calculation. We have also chosen the triplet state using the keyword SPIN. The keyword CIROot has been used to instruct `RASSCF` to find the second excited state in the given symmetry and spin. This is achieved by specifying the number of roots, 1, the dimension of the small CI matrix which must be as large as the highest required root and the number of the required second root. Only for averaged calculations CIROot needs



an additional line containing the weight of the selected roots (unless equal weights are used for all states).

As an alternative to giving inactive and active orbital input we can use the type index input on the ***INPORB*** and indicate there which type the different orbitals should belong to: frozen (f), inactive (i), RAS1 (1), RAS2 (2), RAS3 (3), secondary (s), or deleted (d). This approach is very useful when the input orbitals have been run through `GV`, which is used to select the different subspaces. `GV` will relabel to orbitals according to the users instructions and the corresponding orbital file ,***GvOrb*** can be linked as the ***INPORB*** in the `RASSCF` program without any further input.

A level shift was included using the LEVShift keyword to improve convergence of the calculation. In this case, the calculation does not converge without the use of the level shift. It is advisable to perform new calculations with a non-zero LEVShift value (the default value is 0.5). Another possibility is to increase the maximum number of iterations for the macro and the super-CI Davidson procedures from the default values (200,100) using the keyword ITERations.

Sometimes convergence problems might appear when the wave function is close to fulfill all the convergence criteria. An infrequent but possible divergence might appear in a calculation starting from orbitals of an already converged wave function, or in cases where the convergence thresholds have been decreased below the default values. Option TIGHt may be useful in those cases. It contains the thresholds criteria for the Davidson diagonalization procedure. In situations such as those described above it is recommended to decrease the first parameter of TIGHt to a value lower than the default, for instance 1.0d-06.

`RASSCF` Output

The `RASSCF` section of the *MOLCAS* output contains similar information to the `SCF` output. Naturally, the fact that we have requested an excited state is indicated in the output. In fact, both the lowest triplet state and the first excited state or second root are documented including energies. For both of these states the CI configurations with a coefficient greater than 0.05 are printed along with the partial electron distribution in the active space. Figure 4.10 shows the relevant output for the second root calculated. There are three configurations with a CI-coefficient larger than 0.05 and two with very much larger values. The number of the configuration is given in the first column and the CI-coefficient and weight are given in the last two columns. The electron occupation of the orbitals of the first symmetry for each configuration is given under the '`111`' using '`2`' for a fully occupied orbital and '`u`' for a singly occupied orbital containing an electron with an up spin. The down spin electrons are represented with a '`d`'. The occupation numbers of the active space for each symmetry is given below the contributing configurations. It is important to remember that the active orbitals are not ordered by any type of criterion within the active space.

Figure 4.10: RASSCF portion of output relating to CI configurations and electron occupation of natural orbitals.

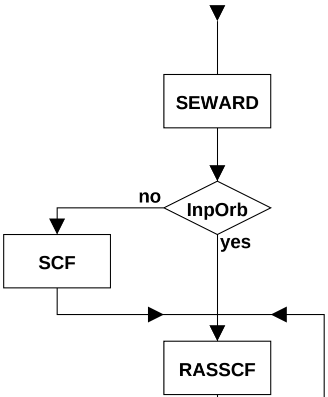
```
printout of CI-coefficients larger than   .05 for root   2
energy=    -75.443990
conf/sym  111 22 4     Coeff  Weight
       3  22u u0 2    .64031  .40999
       4  22u 0u 2    .07674  .00589
      13  2u0 2u 2   -.75133  .56450
      14  2u0 u2 2    .06193  .00384
      19  udu 2u 2    .06489  .00421

Natural orbitals and occupation numbers for root  2
sym 1:   1.986957   1.416217    .437262
sym 2:   1.567238    .594658
sym 4:   1.997668
```

The molecular orbitals are displayed in a similar fashion to the `SCF` section of the output except that the energies of the active orbitals are not defined and therefore are displayed as zero and the electron occupancies are those calculated by the `RASSCF` module. In a state average calculation (more than one root calculated), the MOs will be the natural orbitals corresponding to the state averaged density matrix (called pseudo-natural orbitals) and the occupation numbers will be the corresponding eigenvalues. Natural orbital occupation numbers for each state are printed as shown in Figure 4.10, but the MOs specific to a given state are not shown in the output. They are, however, available in the *`JOBIPH`* file. A number of molecular properties are also computed for the requested electronic state in a similar fashion to the `SCF` module.

Storing and Reading `RASSCF` Orbitals and Wave Functions

The information stored in the `RASSCF` output file, *`JOBIPH`*, can be converted to an editable orbital file using the OUTOrbitals keyword. The type of orbital produced can be either AVERaged, NATUral, CANOnical or SPIN (keywords) orbitals. The OUTOrbitals keyword, combined with the ORBOnly keyword, can be used to read the *`JOBIPH`* file and produce an orbital file, ***`RASORB`***, which can be read by a subsequent `RASSCF` calculation using the same input section. The formatted ***`RASORB`*** file is useful to operate on the orbitals in order to obtain appropriate trial orbitals for a subsequent `RASSCF` calculation. In particular the type index can be changed directly in the file if the `RASSCF` program has converged to a solution with wrong orbitals in the active space. The `RASSCF` program will, however, automatically place the orbital files from the calculation in the user's home directory under the name *`$Project.RasOrb`*, etc. In calculations with spin different from zero the program will also produce the spin orbital files *`$Project.SpdOrb1`*, etc for each state. These orbitals can be used by the program `GV` to produce spin densities.

4.2.5 RASSI — A RAS State Interaction Program



Program `RASSI` (RAS State Interaction) computes matrix elements of the Hamiltonian and other operators in a wave function basis, which consists of individually optimized CI expansions from the `RASSCF` program. Also, it solves the Schrödinger equation within the space of these wave functions. There are many possible applications for such type of calculations. The first important consideration to have into account is that `RASSI` computes the interaction among RASSCF states expanding the same set of configurations, that is, having the same active space size and number of electrons.

The `RASSI` program is routinely used to compute electronic transition moments, as it is shown in section 5.5.1 in the calculation of transition dipole moments for the excited states of the thiophene molecule using CASSCF-type wave functions. By default the program will compute the matrix elements and expectation values of all the operators for which `SEWARD` has computed the integrals and has stored them in the *ONEINT* file.

RASSCF (or CASSCF) individually optimized states are interacting and non-orthogonal. It is imperative when the states involved have different symmetry to transform the states to a common eigenstate basis in such a way that the wave function remains unchanged. The State Interaction calculation gives an unambiguous set of non-interacting and orthonormal eigenstates to the projected Schrödinger equation and also the overlaps between the original RASSCF wave functions and the eigenstates. The analysis of the original states in terms of RASSI eigenstates is very useful to identify spurious local minima and also to inspect the wave functions obtained in different single-root RASSCF calculations, which can be mixed and be of no help to compare the states.

Finally, the `RASSI` program can be applied in situations when there are two strongly interacting states and there are two very different MCSCF solutions. This is a typical situation in transition metal chemistry when there are many close states associated each one to a configuration of the transition metal atom. It is also the case when there are two close quasi-equivalent localized and delocalized solutions. `RASSI` can provide with a single set of orbitals able to represent, for instance, avoided crossings. `RASSI` will produce a number of files containing the natural orbitals for each one of the desired eigenstates to be used in subsequent calculations.

`RASSI` requires as input files the *ONEINT* and *ORDINT* integral files and the *JOBIPH* files from the `RASSCF` program containing the states which are going to be computed. The *JOBIPH* files have to be named consecutively as *JOB001*, *JOB002*, etc. The input for the `RASSI` module has to contain at least the definition of the number of states available in each of the input *JOBIPH* files. Figure 4.12 lists the input file for the `RASSI` program in a calculation including two *JOBIPH* files (2 in the first line), the first one including three roots (3 in the first line) and the second five roots (5 in the first line). Each one of the following lines lists the number of these states within each *JOBIPH* file. Also in the input, keyword NATORB indicates that three files (named sequentially *NAT001*, *NAT002*, and *NAT003*) will be created for the three lowest eigenstates.

Figure 4.11: Sample input requesting the RASSI module to calculate the matrix elements and expectation values for eight interacting RASSCF states

```
␣&RASSI
NROFjobiph
␣2␣3␣5
␣1␣2␣3
␣1␣2␣3␣4␣5
NATOrb
␣3
```

`RASSI` Output

The `RASSI` section of the *MOLCAS* output is basically divided in three parts. Initially, the program prints the information about the *`JOBIPH`* files and input file, optionally prints the wave functions, and checks that all the configuration spaces are the same in all the input states. In second place `RASSI` prints the expectation values of the one-electron operators, the Hamiltonian matrix, the overlap matrix, and the matrix elements of the one-electron operators, all for the basis of input RASSCF states. The third part starts with the eigenvectors and eigenvalues for the states computed in the new eigenbasis, as well as the overlap of the computed eigenstates with the input RASSCF states. After that, the expectation values and matrix elements of the one-electron operators are repeated on the basis of the new energy eigenstates. A final section informs about the occupation numbers of the natural orbitals computed by `RASSI`, if any.

In section 5.5.1 a detailed example of how to interpret the matrix elements output section for the thiophene molecule is displayed. The rest of the output is self-explanatory. It has to be remembered that to change the default origins for the one electron operators (the dipole moment operator uses the nuclear charge centroid and the higher order operators the center of the nuclear mass) keyword CENTER in `SEWARD` must be used. Also, if multipoles higher than order two are required, the option MULTIPOLE has to be used in `SEWARD`.

The program `RASSI` can also be used to compute a spin-orbit Hamiltonian for the input CASSCF wave functions as defined above. The keyword AMFI has to be used in `SEWARD` to ensure that the corresponding integrals are available.

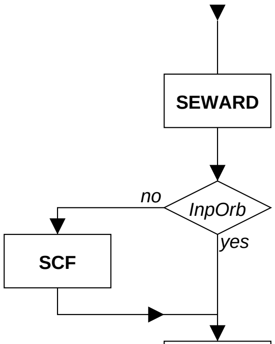
Figure 4.12: Sample input requesting the RASSI module to calculate and diagonalize the spin-orbit Hamiltonian the ground and triplet excited state in water.

```
␣&RASSI
NROFjobiph
␣2␣1␣1
␣1
␣1
spinorbit
Ejob
```

The first *`JOBMIX`* file contains the wave function for the ground state and the second file the $^3B_2$ state discussed above. The keyword EJOB makes the `RASSI` program use the CASPT2 energies which have been written on the *`JOBMIX`* files in the diagonal of the spin-orbit Hamiltonian. The output of this calculation will give four spin-orbit states and the corresponding transition properties, which can for example be used to compute the radiative lifetime of the triplet state.

4.2.6 CASPT2 — A Many Body Perturbation Program

Dynamic correlation energy of a molecular system can be calculated using the `CASPT2` program module in *MOLCAS*. A `CASPT2` calculation gives a second order perturbation estimate of the full CI energy using the CASSCF wave function of the system. As can be seen in the flowchart, `CASPT2` follows a `RASSCF` calculation but in this



case the `RASSCF` calculation must produce a CASSCF wave function. The program can also performe Multi-State CASPT2 caclulations (MS-CASPT2) in which different CASPT2 states are coupled using an effective Hamiltonian computed to second order in perturbatrin theory. This is necessary in cases where different CASSCF wave functions are strongly dependent on dynamical correlation effects. The wave function have to be obtained in a previous State-Average CASSCF calculation.

A sample input is given in Figure 4.13. The FROZen keyword specifies the number of orbitals of each symmetry which will not be included in the correlation. We have chosen the `RASSCF` INACtive orbitals to be frozen for this calculation (the default is to freeze all core orbitals, so the input is strictly not needed). The remaining two keywords, CONVergence and MAXIter, are included with there default values. The MULTistate is included fo clarity even if not needed in this single state calculation. A single line follows indicating the number of simultaneously treated CASPT2 roots and the number of the roots in the previous SA-CASSCF calculation.

`CASPT2` Output

In section 5.5.1 the meaning and significance of most of the features used and printed by the `CASPT2` program are explained in the context of an actual example. We suggest a careful reading of that section because understanding the results of a CASPT2 calculation is important for the analysis of problems like intruder states, large coefficients, convergence, etc.

Figure 4.13: Sample input requesting the CASPT2 module to calculate the CASPT2 energy of a water molecule in C_{2v} symmetry with one frozen orbital.

```
␣&CASPT2
Frozen
␣1␣0␣0␣0
Convergence
␣1.0e-07
Multistate
1␣␣1
MaxIter
␣40
```

The output of the `CASPT2` program begins with the title from the input as well as the title from the `SEWARD` input. It also contains the cartesian coordinates of the molecule and the CASSCF wave function and orbital specifications. This is followed by details about the type of Fock and H_0 operator used and, eventually, the value of the level-shift parameter employed. It is possible then to obtain, by input specifications, the quasi-canonical orbitals in which the wave function will be represented. The following CI vector and occupation number analysis will be performed using the quasi-canonical orbitals.

Two important sections follow. First a detailed report on small energy denominators, large components, and large energy contributions which will inform about the reliability of the calculation (see section 5.5.1) and finally the `CASPT2` property section including the natural orbitals obtained as defined in the output and a number of approximated molecular properties.

If the Multistate option is used, the program will perform one CASPT2 calculation for each one of the selected roots, and finally the complete effective Hamiltonian containing the selected states will be solved to obtain the final MS-CASPT2 energies and PM-CASSCF wave functions [13].

The `CASPT2` module needs the integral files in $WorkDir and the *`RUNFILE`* file from the and the *`JOBIPH`* file from the `RASSCF` module. The orbitals are saved in the *`PT2ORB`* file. The new PM-CASSCF wave functions generated in a MS-CASPT2 calculation is saved in the *`JOBMIX`* file.

4.2.7 CASVB — A non-orthogonal MCSCF program

`CASVB` is a program for carrying out quite general types of non-orthogonal MCSCF calculations, offering, for example, all the advantages associated with working within a valence bond formalism.

Warning: as for any general MCSCF program, one may experience convergence problems, (*e.g.,* due to redundant parameters), and the non-orthogonal optimization of orbitals can furthermore give linear dependency problems. Several options in `CASVB` can help overcoming these difficulties.

This program can be used in two basic modes:

a) fully variational optimization

b) representation of CASSCF wavefunctions using overlap- (*relatively inexpensive*) or energy-based criteria.

`CASVB` executes the following logical steps: Setup of wavefunction information, starting guess generation, one, or several, optimization steps, various types of analysis of the converged solution.

`CASVB` input

`CASVB` attempts to define defaults for as many input quantities as possible, so that in the simplest case no input to the `CASVB` module is required. Sample input for a CASVB calculation on the lowest singlet state of CH_2:

```
␣&GATEWAY
coord
ch2.xyz
group
x␣y
basis␣set
sto-3g....
␣&SEWARD
␣&SCF
occupied
3␣0␣1␣0
```

```
␣&RASSCF
inactive
1␣0␣0␣0
ras2
3␣1␣2␣0
nactel
6␣0␣0
lumorb
␣&CASVB
```

`CASVB` output

The amount of output in `CASVB` depends heavily on the setting of the PRINT levels. In case of problems with convergence behaviour it is recommended to increase these from their rather terse default values.

In the following the main features of the output are outlined, exemplified by the job in the input above. Initially, all relevant information from the previous `RASSCF` calculation is recovered from the *`JOBIPH`* interface file, after which the valence bond wavefunction information is summarized, as shown below. Since spatial configurations have not been specified explicitly in this example, a single covalent configuration is chosen as default. This gives 5 spin-adapted VB structures.

```
Number of active electrons :   6
          active orbitals  :   6
          Total spin       : 0.0
          State symmetry   :   1

Spatial VB configurations
-------------------------
    Conf. =>   Orbitals
      1   =>    1  2  3  4  5  6

Number of VB configurations :     1
          VB structures     :     5
          VB determinants   :    20
```

The output from the following optimization steps summarizes only the most relevant quantities and convergence information at the default print level. For the last optimization step, for example, The output below thus states that the VB wavefunction was found by maximizing the overlap with a previously optimized CASSCF wavefunction (output by the `RASSCF` program), and that the spin adaptation was done using the Yamanuchi-Kotani scheme. Convergence was reached in 7 iterations.

```
-- Starting optimization - step  3 --------

Overlap-based optimization (Svb).

Optimization algorithm:            dFletch
Maximum number of iterations:           50
Spin basis:                         Kotani

------------------------------------------
Optimization entering local region.
```

```
Converged ... maximum update to coefficient:  0.59051924E-06
Final Svb :    0.9978782695
Number of iterations used:   7
```

Finally in the output below the converged solution is printed; orbital coefficients (in terms of the active CASSCF MOs) and structure coefficients. The overlap between orbitals are generally of interest, and, as also the structures are non-orthogonal, the structure weights in the total wavefunction. The total VB wavefunction is not symmetry-adapted explicitly (although one may ensure the correct symmetry by imposing constraints on orbitals and structure coefficients), so its components in the various irreducible representations can serve to check that it is physically plausible (a well behaved solution generally has just one non-vanishing component).

Next follows the one-electron density with natural-orbital analysis, again with quantities printed in the basis of the active CASSCF MOs.

```
Orbital coefficients :
----------------------
          1           2           3           4           5           6
  1  0.43397359 -0.43397359 -0.79451779 -0.68987187 -0.79451780 -0.68987186
  2 -0.80889967  0.80889967 -0.05986171 -0.05516284 -0.05986171 -0.05516284
  3  0.00005587 -0.00005587  0.20401015 -0.20582094  0.20401016 -0.20582095
  4  0.39667145  0.39667145  0.00000000  0.00000000  0.00000000  0.00000000
  5 -0.00000001 -0.00000001 -0.53361427 -0.65931951  0.53361425  0.65931952
  6  0.00000000  0.00000000  0.19696124 -0.20968879 -0.19696124  0.20968879

Overlap between orbitals :
--------------------------
          1           2           3           4           5           6
  1  1.00000000 -0.68530352 -0.29636622 -0.25477647 -0.29636623 -0.25477647
  2 -0.68530352  1.00000000  0.29636622  0.25477647  0.29636623  0.25477646
  3 -0.29636622  0.29636622  1.00000000  0.81994979  0.35292419  0.19890631
  4 -0.25477647  0.25477647  0.81994979  1.00000000  0.19890634  0.04265679
  5 -0.29636623  0.29636623  0.35292419  0.19890634  1.00000000  0.81994978
  6 -0.25477647  0.25477646  0.19890631  0.04265679  0.81994978  1.00000000

Structure coefficients :
------------------------
     0.00000000  0.00000001  0.09455957  0.00000000 -0.99551921

Saving VB wavefunction to file VBWFN.

Saving VB CI vector to file JOBIPH.

Svb :          0.9978782695
Evb :        -38.4265149062

Chirgwin-Coulson weights of structures :
----------------------------------------
VB spin+space (norm   1.00000000) :
     0.00000000  0.00000000 -0.00211737  0.00000000  1.00211737
VB spin only  (norm   0.38213666) :
     0.00000000  0.00000000  0.00894151  0.00000000  0.99105849

Symmetry contributions to total VB wavefunction :
-------------------------------------------------
Irreps 1 to 4 :  0.10000000E+01  0.15118834E-17  0.17653074E-17  0.49309519E-17
```

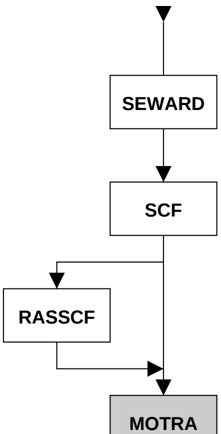
```
Energies for components > 1d-10 :
---------------------------------
Irreps 1 to 4 : -0.38426515E+02  0.00000000E+00  0.00000000E+00  0.00000000E+00

One-electron density :
----------------------
          1           2           3           4           5           6
  1  1.98488829 -0.00021330  0.00011757  0.00000000  0.00000000  0.00000000
  2 -0.00021330  1.90209222 -0.00006927  0.00000000  0.00000000  0.00000000
  3  0.00011757 -0.00006927  0.02068155  0.00000000  0.00000000  0.00000000
  4  0.00000000  0.00000000  0.00000000  0.09447774  0.00000000  0.00000000
  5  0.00000000  0.00000000  0.00000000  0.00000000  1.97572540 -0.00030574
  6  0.00000000  0.00000000  0.00000000  0.00000000 -0.00030574  0.02213479

Natural orbitals :
------------------
          1           2           3           4           5           6
  1 -0.99999668  0.00000000  0.00257629  0.00000000  0.00000000  0.00005985
  2  0.00257628  0.00000000  0.99999668  0.00000000  0.00000000 -0.00003681
  3 -0.00005995  0.00000000 -0.00003666  0.00000000 -0.00000001 -1.00000000
  4  0.00000000  0.00000000  0.00000000  1.00000000  0.00000001  0.00000000
  5  0.00000000  0.99999999  0.00000000  0.00000000  0.00015650  0.00000000
  6  0.00000000 -0.00015650  0.00000000 -0.00000001  0.99999999 -0.00000001

Occupation numbers :
--------------------
          1           2           3           4           5           6
  1  1.98488885  1.97572545  1.90209167  0.09447774  0.02213475  0.02068154
```

Viewing and plotting VB orbitals

In many cases it can be helpful to view the shape of the converged valence bond orbitals. *MOLCAS* therefore provides two facilities for doing this. For the Molden program, an interface file is generated at the end of each `CASVB` run (see also Section ??). Alternatively a `CASVB` run may be followed by `RASSCF` (Section 8.30) and `GRID_IT` (Section 8.16) with the VB specification, in order to generate necessary files for viewing with `GV`.

4.2.8 MOTRA — An Integral Transformation Program

Integrals saved by the `SEWARD` module are stored in the Atomic Orbital (AO) basis. Some programs have their own procedures to transform the integrals into the Molecular Orbital (MO) basis. The *MOLCAS* `MOTRA` module performs this task for Configuration Interaction (CI), Coupled- and Modified Coupled-Pair (CPF and MCPF, respectively) and Coupled-Cluster (CC) calculations.



The sample input below contains the `MOTRA` input information for our continuing water calculation. We firstly specify that the `RASSCF` module interface file will be the source of the orbitals using the keyword JOBIPH. The keyword FROZEN is used to specify the number of orbitals in each symmetry which will not be correlated in subsequent calculations. This can also be performed in the corresponding `MRCI`, `CPF` or CC programs but is more efficient to freeze them here. Virtual orbitals can be deleted using the DELETE keyword.

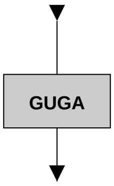
```
␣&MOTRA
JobIph
Frozen
␣1␣0␣0␣0
*Delete
*␣1␣0␣0␣0
```

`MOTRA` Output

The `MOTRA` section of the output is short and self explanatory. The integral files produced by `SEWARD`, *`ONEINT`* and *`ORDINT`*, are used as input by the `MOTRA` module which produces the transformed symbolic files *`TRAONE`* and *`TRAINT`*, respectively. In our case, the files are called *`water.TraOne`* and *`water.TraInt`*, respectively.

The `MOTRA` module also requires input orbitals. If the LUMOrb keyword is specified the orbitals are taken from the *`INPORB`* file which can be any formated orbital file such as *`water.ScfOrb`* or *`water.RasOrb`*. The JOBIph keyword causes the `MOTRA` module to read the required orbitals from the *`JOBIPH`* file.

4.2.9 GUGA — A Configuration Interaction Coupling Coefficients Program



Several of the Configuration Interaction (CI) modules in *MOLCAS* use the `GUGA` module to compute the CI coupling coefficients. We continue our water calculations using the input file shown in the input below. The TITLe keyword behaves in a similar fashion as described in previous modules. There are several compulsory keywords of the `GUGA` module. The number of electrons to be correlated is specified using the ELECtrons keyword. We are correlating the valence electrons. The spin state is specified using the SPIN keyword.

Sample input requesting the the GUGA module to calculate the coupling coefficients for neutral triplet water in C_{2v} symmetry with six electrons in the active space:

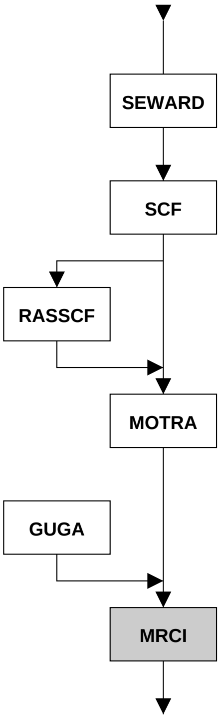
```
␣&GUGA
Title
␣␣GUGA␣for␣C2v␣Water
Electrons
␣8
Spin
␣3
CIAll
␣1
Inactive
␣1␣0␣0␣0
Active
␣2␣2␣0␣1
```

The keywords CIALl and REFErence are mutually exclusive. We specify CIALl which will calculate the energy using all possible references functions that can be constructed using the input set of occupation numbers of the active orbitals regardless of the spin coupling (all configurations used to build the corresponding CASSCF wave function). Specific selected references can be chosen using the REFErence keyword. Either the ACTIve or INACtive keyword should be used for a meaningful calculation. The default for both keywords is zero for all symmetries. These keywords function in a similar fashion to these in the `RASSCF` program module. The INACTIVE keyword specifies the orbitals that are fully occupied in each symmetry in all the reference functions and the ACTIVE keyword specifies the orbitals that may have varying occupations in all references. The selection of INACTIVE orbitals in the input above is forcing the bonding *sp* hybrid orbital to remain fully occupied in all reference states.

`GUGA` Output

The `GUGA` section of the output lists the possible configurations in the active space. There are nine possible triplet configurations of six electrons in five orbitals. Apart from the various types of orbital in each symmetry the `GUGA` section of the output also gives the number of states that will coupled with various states. There are no input files for the `GUGA` module but the calculated coupling coefficients are stored in *`CIGUGA`*.

4.2.10 MRCI — A Configuration Interaction Program



Multi Reference Single and Doubles Configuration Interaction (MR-SDCI) wave functions are produced by the `MRCI` program module in the *MOLCAS* codes. The SDCI keyword requests an ordinary Multi Reference Single and Doubles Configuration Interaction calculation. This is the default and is mutually exclusive with the ACPF keyword which requests an Average Coupled Pair Function calculation. The final keyword, ROOT, specifies the number of the CI root the calculation should compute. The second CI root is the first excited state and since the `GUGA` module has computed the coupling coefficients for a triplet state, the `MRCI` module will converge to the first excited triplet state.

`MRCI` Output

The `MRCI` section of the output lists the number of each type of orbital in each symmetry including pre-frozen orbitals that were frozen by the `GUGA` module. There is a list of the reference configurations with the inactive orbitals included. An empty orbital is listed as '`0`' and a doubly occupied as '`3`'. The spin of a singly occupied orbital by '`1`' (spin up) or '`2`' (spin down). The total number of configuration state functions (CSFs) is listed below the reference configurations.

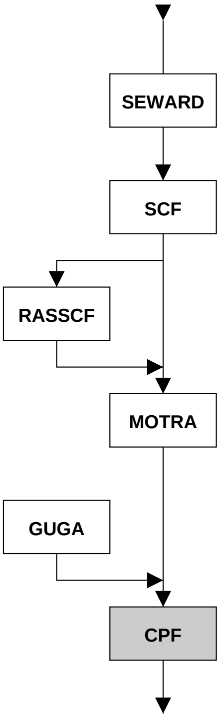
Sample input requesting the the MRCI module to calculate the first excited MRCI energy for neutral triplet water in C_{2v} symmetry with six electrons in the active space:

```
 &MRCI
Title
 MR-SDCI of 2nd CI root of C2v Water
SDCI
Root
 2
```

A listing of the possible CI roots is followed by the CI iteration and convergence information. The Davidson and ACPF corrections are included along with the important CSFs in the CI wave function. The molecular orbitals are listed near the end of the output.

There are four input files to the `MRCI` module; *`CIGUGA`* from `GUGA`, *`TRAONE`* and *`TRAINT`* from `MOTRA` and *`ONEINT`* from `SEWARD`. The orbitals are saved in *`CIORBnn`* where nn is the number of the CI root.

4.2.11 CPF — A Coupled-Pair Functional Program



The `CPF` program produces Single and Doubles Configuration Interaction (SDCI), Coupled-Pair Functional (CPF), Modified Coupled-Pair Functional (MCPF), and Averaged Coupled-Pair Functional (ACPF) wave functions (see CPF section of the user's guide) from one reference configuration. The difference between the `MRCI` and `CPF` codes is that the former can handle Configuration Interaction (CI) and Averaged Coupled-Pair Functional (ACPF) calculations with more than one reference configuration. For a closed-shell reference the wave function can be generated with the `SCF` program. In open-shell cases the `RASSCF` has to be used.

The TITLE keyword behaviors in a similar fashion to the other MOLCAS modules. The CPF keyword requests an Coupled-Pair Functional calculation. This is the default and is mutually exclusive with keywords MCPF, ACPF, and SDCI which request different type of calculations. The input below lists the input files for the `GUGA` and `CPF` programs to obtain the MCPF energy for the lowest triplet state of B_2 symmetry in the water molecule. The `GUGA` module computes the coupling coefficients for a triplet state of the appropriate symmetry and the `CPF` module will converge to the first excited triplet state. One orbital of the first symmetry has been frozen in this case (core orbital) in the `MOTRA` step.

`CPF` Output

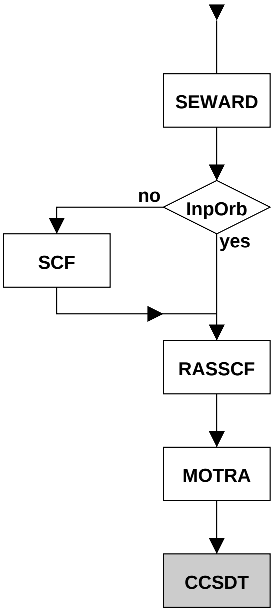
The `CPF` section of the output lists the number of each type of orbital in each symmetry including pre-frozen orbitals that were frozen by the `GUGA` module. After some information concerning the total number of internal configurations used and storage data, it appears the single reference configuration in the `MRCI` format: an empty orbital is listed as '`0`' and a doubly occupied as '`3`'. The spin of a singly occupied orbital by '`1`' (spin up) or '`2`' (spin down). The molecular orbitals are listed near the end of the output.

Sample input requested by the GUGA and CPF modules to calculate the ACPF energy for the lowest B_1 triplet state of the water in C_{2v} symmetry:

```
␣&GUGA
Title
␣H2O␣molecule.␣Triplet␣state.
Electrons
␣8
Spin
␣3
Inactive
␣2␣␣␣␣0␣␣␣␣1␣␣␣␣0
Active
␣1␣␣␣␣1␣␣␣␣0␣␣␣␣0
CiAll
␣2
␣&CPF
Title
␣MCPF␣of␣triplet␣state␣of␣C2v␣Water
MCPF
```

There are four input files to the `CPF` module; *`CIGUGA`* from `GUGA`, *`TRAONE`* and *`TRAINT`* from `MOTRA` and *`ONEINT`* from `SEWARD`. The orbitals are saved in *`CPFORB`*.

4.2.12 CCSDT — A Set of Coupled-Cluster Programs



The *MOLCAS* program `CCSDT` is really a shell script which calls sequentially to a set of three programs, which compute Coupled-Cluster Singles Doubles, CCSD, and Coupled-Cluster Singles Doubles and Non-iterative Triples Correction CCSD(T) wave functions for restricted single reference both closed- and open-shell systems. The set is composed by three modules: program `CCSORT` performs a reorganization of the integrals and the reference function from previous runs; program `CCSD` computes the CCSD wave function and energy allowing for different forms of spin adaptation, and program `CCT3` computes the perturbative triples correction for the CCSD wave function in the different approaches explained in the CCSDT section of the user's guide.

There are two possibilities to run the programs. One is to use the command molcas $Input, where the input for the program must be placed in file *$Input*. Other possibility is run the programs sequentially: molcas ccsort.input, molcas ccsd.input, and molcas cct3.input. The final possibility is to use `AUTOMOLCAS`. In any case the programs are run sequentially: first `CCSORT`, second `CCSD`, and, if required, `CCT3`.

In addition to the *ONEINT* and *ORDINT* integral files, the CCSD(T) codes require the *JOBIPH* file containing the reference wave function (remember that it is not possible to compute open-shell systems with the `SCF` program) and the transformed two-electron integrals produced by the `MOTRA` module and stored in the *TRAINT* file.

Previously to execute the `CCSORT` module, wave functions and integrals have to be prepared. First, a RASSCF calculation has to be run in such a way that the resulting wave function has one single reference. In closed-shell situations this means to include all the orbitals as inactive and set the number of active electrons to zero. Keyword OUTORBITALS followed by the specification CANONICAL must be used in the `RASSCF` input to activate the construction of canonical orbitals and the calculation of the CI-vectors on the basis of the canonical orbitals. After that the `MOTRA` module has to be run to transform the two-electron integrals using the molecular orbitals provided by the `RASSCF` module. If the LUMORB is used in the `MOTRA` input it will be necessary to run a previous `RASREAD` program using the option CANONICAL in the `RASREAD` input. Otherwise, the *JOBIPH* from the `RASSCF` calculation can be used directly by `MOTRA` using the JOBIPH option in the `MOTRA` input. Frozen or deleted orbitals can be introduced in the transformation step by the proper options in the `MOTRA` input.

`CCSORT`, `CCSD`, and `CCT3` Outputs

The section of the *MOLCAS* output corresponding to the CC programs is self explanatory. The default `CCSORT` output simply contains the wave function specifications from the previous RASSCF calculation, the orbital specifications, and the diagonal Fock matrix elements and orbital energies. The default `CCSD` output contains the technical description of the calculation, the iterations leading to the CCSD energy, and the five largest amplitudes of each type, which will help to evaluate the calculation. The default `CCT3` output contains the description of the employed method (from the three available) to compute perturbatively the triple excited contributions to the CC energy, the value of the correction, and the energy decomposition into spin parts.

Example of a CCSD(T) calculation

Figure 4.14 contains the input files required by the `SEWARD`, `SCF`, `RASSCF`, `MOTRA` and `CCSDT` programs to compute the ground state of the HF^+ cation. molecule, which is a doublet of Σ^+ symmetry. A more detailed description of the different options included in the input of the programs can be found in the CCSDT section of the user's guide. This example describes how to calculate CCSD(T) energy for HF(+) cation. This cation can be safely represented by the single determinant as a reference function, so one can assume that CCSD(T) method will be suitable for its description.

The calculation can be divided into few steps:

1. Run `SEWARD` to generate AO integrals.
2. Calculate the HF molecule at the one electron level using `SCF` to prepare an estimate of MO for the `RASSCF` run.
3. Calculate HF(+) cation by subtracting one electron from the orbital with the first symmetry. There is only one electron in one active orbital so only one configuration is created. Hence, we obtain a simple single determinant ROHF reference.
4. Perform MO transformation exploiting `MOTRA` using MO coefficients from the `RASSCF` run.
5. Perform the Coupled Cluster calculation using `CCSDT` program. First, the data produced by the programs `RASSCF` and `MOTRA` need to be reorganized, then the CCSD calculation follows, with the chosen spin adaptation being T2 DDVV. Finally, the noniterative triple excitation contribution calculation is following, where the CCSD amplitudes are used.

This is an open shell case, so it is suitable to choose CCSD(T) method as it is defined by Watts *et al.* [14]. Since CCSD amplitudes produced by previous `CCSD` run are partly spin adapted and denominators are produced from the corresponding diagonal Fock matrix elements, final energy is sometimes referred as SA1 CCSD$(T)_d$ (see [15]).

Figure 4.14: Sample input containing the files required by the SEWARD, SCF, RASSCF, MOTRA, CCSORT, CCSD, and CCT3 programs to compute the ground state of the HF^+ cation.

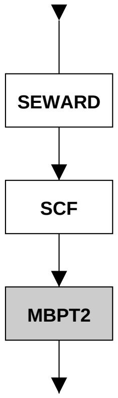
```
␣&SEWARD␣&END
Title
␣HF␣molecule
Nopack
Symmetry
X␣Y
Basis␣set
F.ano-l...3S2P1D.
F␣␣␣␣␣␣0.00000␣␣␣0.00000␣␣␣1.73300
End␣of␣basis
```

```
Basis␣set
H.ano-l...2S1P.
H␣␣␣␣␣␣0.00000␣␣␣0.00000␣␣␣0.00000
End␣of␣basis
End␣of␣input


␣&SCF␣&END
Title
␣HF␣molecule
Occupied
␣3␣1␣1␣0
End␣of␣input


␣&RASSCF␣&END
Title
␣HF(+)␣cation
OUTOrbitals
␣Canonical
Symmetry
␣1
Spin
␣2
nActEl
␣1␣0␣0
Inactive
␣2␣1␣1␣0
Ras2
␣1␣0␣0␣0
LumOrb
End␣of␣input


␣&MOTRA␣&END
Title
␣HF(+)␣cation
JobIph
Frozen
␣1␣0␣0␣0
End␣of␣input


␣&CCSDT␣&END
Title
␣HF(+)␣cation
Iterations
50
Denominators
2
Shift
0.2,0.2
Accuracy
1.0d-7
Adaptation
1
Extrapolation
5,4
Triples
3
T3Denominators
0
End␣of␣input
```

`RASSCF` calculates the HF ionized state by removing one electron from the orbital in the first symmetry. Do not forget to use keyword CANONICAL. In the `CCSDT` run, the number of iterations is limited to 50. Denominators will be formed using orbital energies. (This corresponds to the chosen spin adaptation.) Orbitals will be shifted by 0.2 a.u., what will accelerate the convergence. However, final energy will not be affected by the chosen type of denominators and orbital shifts. Required accuracy is 1.0d-7 a.u. for the energy. T2 DDVV class of CCSD amplitudes will be spin adapted. To accelerate the convergence, DIIS procedure is exploited. It will start after 5th iteration and the last four iterations will be taken into account in each extrapolation step.

In the triples step the CCSD(T) procedure as defined by Watts *et al.* [14] will be performed. Corresponding denominators will be produced using diagonal Fock matrix elements.

4.2.13 MBPT2 — A Second-Order Many-Body PT RHF Program



The `MBPT2` program performs second-order Many Body Perturbation Theory calculations based on a RHF-type of wave function (MP2 method). The calculation is to some extent defined by the SCF calculation which must be performed before running the `MBPT2` program. Therefore, there is no difficulty related to the input file unless an analysis of the correlation energies of specific electron pairs or contribution from external orbitals wants to be performed. In this case keywords SFROzen and SDELeted have to be used as described in section 8.22 of the user's guide.

To run the program the *`ORDINT`* integral file(s) generated by the `SEWARD` program and the *`RUNFILE`* file generated by the `SCF` program are needed. The program can be otherwise run in a direct manner. Therefore the `SEWARD` program can be run with the option DIREct included in its input. Only the *`ONEINT`* will then be generated and used by the `SCF` module. The input file used to run an `MBPT2` calculation on the ground state of the water molecule is displayed in figure 4.15. For large molecules it is also possible to use the Cholesky decomposition technique to speed up the calculations. This will be described in another section of the tutorials.

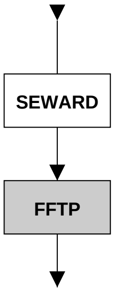
```
␣&MBPT2
Title
␣MP2␣of␣ground␣state␣of␣C2v␣Water␣with␣the␣1s␣orbital␣on␣oxygen␣left
␣uncorrelated.
Frozen
␣1␣0␣0␣0
```

Figure 4.15: Sample input requested by the MBPT2 module to calculate the MP2 energy for the ground state of the water in C_{2v} symmetry.

The output of `MBPT2` is self-explanatory.

4.2.14 FFPT — A Finite Field Perturbation Program

Many molecular properties of wave functions can be computed using the `FFPT` program module in *MOLCAS*. It adds the requested operator to the integrals computed by the `SEWARD` module. This must be done before the *MOLCAS* module calculating the required wave function is requested so the `FFPT` module is best run directly after the `SEWARD` module.



The TITLE keyword behaviors in a similar fashion to other *MOLCAS* modules. The sample input below contains the `FFPT` input requesting that the dipole moment operator be added to the integrals using the DIPOLE keyword. The size and direction is specified using the COMP keyword which accepts free format input. We can compute the dipole of the molecule by numerical determination of the gradient of the energy curve determined for several values of the dipole operator. From the second derivative we can obtain the polarizability component.

Sample input requesting the FFPT module to include a dipole moment operator in the integral file:

```
␣&FFPT
Title
␣Finite␣Perturbation␣calculation␣using␣a␣dipole
␣in␣the␣x␣negative␣direction␣of␣strength␣0.1␣au
FFPT
Dipole
␣Comp
␣X␣-0.1
```

`FFPT` Output

The `FFPT` section of the output is short and self explanatory. The *`ONEINT`* file is updated with the requested operator.

4.2.15 VIBROT — A Program for Vibration-Rotation on Diatomic Molecules

The program `VIBROT` computes vibration-rotation spectra for diatomic molecules. As input it uses a potential curve computed pointwise by any of the wave function programs. It does not require other input file from any of the *MOLCAS* programs, just its standard input file.

In section 5.1.2 the reader will find an overview of the input and output files required by `VIBROT` and the different uses of the program on the calculation of the electronic states of the C_2 molecule. The reader is referred to section 5.1.2 and section 8.35 of the user's guide for a detailed description of the program.

4.2.16 GENANO — A Program to Generate ANO Basis Sets

`GENANO` is a program for determining the contraction coefficients for generally contracted basis sets. They are determined by diagonalizing a density matrix, using the eigenvectors (natural orbitals) as the contraction coefficients, resulting in basis sets of the ANO (Atomic Natural Orbitals) type. The program can be used to generate any set of atomic or molecular basis functions. Only one or more wave functions (represented by formated orbital files) are needed to generate the average density matrix. These natural orbital files can be produced by any of the wave function generators, as it is described in section 8.15 of the user's guide. As an illustrative example, in section 5.5.1 there is an example of how to generate a set of molecular basis set describing Rydberg orbitals for the benzene molecule. The reader is referred to this example for more details.

The `GENANO` program requires several input files. First, one *`ONEINT`* file generated by the `SEWARD` module for each input wave function. The files must be linked as *`ONE001`*, *`ONE002`*, etc. If the wave functions correspond to the same system, the same *ONEINT* file must be linked with the corresponding names as many times as wave functions are going to be treated. Finally, the program needs one file for wave function containing the formated set of natural orbitals. The files must be linked as *NAT001*, *NAT002*, etc.

The input file for module `GENANO` contains basically three important keywords. CENTER defines the atom label for which the basis set is to be generated. The label must match the label it has in the `SEWARD`. SETS keyword indicates that the next line of input contains the number of sets to be used in the averaging procedure and WEIGHTS defines the relative weight of each one of the previous sets in the averaging procedure. Figure 4.16 lists the input file required by the `GENANO` program for making a basis set for the oxygen atom. Three natural orbital files are expected, containing the natural orbitals for the neutral atom, the cation, and the anion.

Figure 4.16: Sample input requesting the GENANO module to average three sets of natural orbitals on the oxygen atom.

```
␣&GENANO
Title
␣Oxygen␣atom␣basis␣set:␣O/O+/O-
Center
␣O
Sets
␣3
Weights
␣0.50␣0.25␣0.25
End␣Of␣Input
```

As output files `GENANO` provides the file *ANO*, containing the contraction coefficient matrix organized such that each column correspond to one contracted basis function, and the file *FIG*, which contains a PostScript figure file of the obtained eigenvalues. The output of `GENANO` is self-explanatory.

4.2.17 ALASKA — A Program for Integral Derivatives

`ALASKA` computes the first derivatives of the one- and two-electron integrals with respect to the nuclear displacements. The derivatives are contracted with the one- and two-electron densities to form the molecular gradients, which will be used by the program `SLAPAF`. At present the `ALASKA` module computes SCF/DFT and MCSCF gradients analytically, the rest are computed numerically. The `ALASKA` module is automatically envoked when needed if the user has not explicitly requested the module to be executed. We postpone the discussion about `ALASKA` to section 4.2.21.

4.2.18 SLAPAF — A Program for Geometry Optimizations, Transition States, etc.

Program `SLAPAF` is tailored to use analytical or numerical gradients produced by `ALASKA` to relax the geometry of a molecule towards an energy minimum or a transition state. The program is also used for finding inter state crossings (ISC), conical itersections (CI), to compute reaction paths, intrinsic reaction coordinate (IRC) paths, etc. We postpone the discussion about `SLAPAF` to section 4.2.21.

4.2.19 MCKINLEY — A Program for Integral Second Derivatives

`MCKINLEY` computes the analytic second derivatives of the one- and two-electron integrals with respect to the nuclear positions at the SCF and CASSCF level of theory. The differentiated integrals can be used by program `MCLR` to performs response calculations on single and multiconfigurational SCF wave functions. One of the basic uses of `MCKINLEY` and `MCLR` is to compute analytical hessians (vibrational frequencies, IR intensities, etc). Note that `MCKINLEY` for a normal frequency calculations will automatically start the `MCLR` module! For all other methods a numerical procedure is automatically envoked by `MCKINLEY` to compute the vibrational frequencies.

4.2.20 MCLR — A Program for Linear Response Calculations

`MCLR` computes response calculations on single and multiconfigurational SCF wave functions. One of the basic uses of `MCKINLEY` and `MCLR` is to compute analytical hessians (vibrational frequencies, IR intensities, etc). `MCLR` can also calculate the Lagrangian multipliers for a MCSCF state included in a state average optimization and construct the effective densities required for analytical gradients of such a state. The use of keyword RLXROOT in the `RASSCF` program is required. In both cases the explicit request of executing the `MCLR` module is not required and will be automatic. We postpone further discussion about `MCLR` to section 4.2.21.

It follows an example of how to compute the analytical hessian of an excited state:

```
␣&SEWARD
Title=p-benzoquinone␣anion.␣Casscf␣optimized␣geometry.
Symmetry=␣X␣Y␣Z
Basis␣set
C.ANO-L...4s3p2d.
␣␣␣C1␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣2.2783822672␣␣␣␣␣␣␣1.3271399214
␣␣␣C2␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣2.7374556550
end␣of␣basis
Basis␣set
H.ANO-L...3s2p.
␣␣␣H1␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣4.0361650878␣␣␣␣␣␣␣2.3432668589
End␣of␣basis
Basis␣set
O.ANO-L...4s3p2d.
␣␣␣O1␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣5.1965257318
End␣of␣basis
␣&SCF
TITLE=p-benzoquinone(-)␣D2h
OCCUPIED=8␣2␣5␣1␣7␣1␣4␣0;␣ITERATIONS=␣40
␣&RASSCF
LUMORB
TITLE=p-benzoquinone␣anion.␣2B3u␣state.
SYMMETRY=2;␣SPIN=2;␣NACTEL=9␣0␣0
INACTIVE=8␣␣0␣␣5␣␣0␣␣7␣␣0␣␣4␣␣0
RAS2␣␣␣␣=0␣␣3␣␣0␣␣1␣␣0␣␣3␣␣0␣␣1
CIROOT=␣1␣1;␣1
ITER=50,25
␣&MCKINLEY;␣NoMCLR
␣&MCLR;␣Thre=0.0001;␣Print=255;␣Rassi
```

Since we do not want the `MCKINLEY` module to start the `MCLR` module automatically with the default input we have added the keyword NOMCLR to the `MCKINLEY` input and provided our own `MCLR` input! The keyword RASSI, is used to transform the CI vectors to split GUGA representation and the orbital rotations to AO basis to make the response accessible for state interaction calculations, it allows to compute the transition dipole moment geometry derivatives for further uses.

4.2.21 A Molecular Structure Optimization

One of the most powerful functions of *ab initio* calculations is geometry predictions. The minimum energy structure of a molecule for a given method and basis set is instructive especially when experiment is unable to determine the actual geometry. MOLCAS performs a geometry optimization with analytical gradients at the SCF or RASSCF level of calculation, and with numerical gradients at the CASPT2 level.

In order to perform geometry optimization an input file must contain a loop, which includes several calls: calculation of integrals (SEWARD), calculation of energy (SCF, RASSCF, CASPT2), calculation of gradients (ALASKA), and calculation of the new geometry (SLAPAF).

This is an example of such input

```
>> set MaxIter 25 <<
>> Do While  <<
  &GATEWAY
   coord
    file.xyz
   basis ANO-S-MB
  &SEWARD
  &SCF
  &SLAPAF
>> EndDo <<
```

The initial coordinates will be taken from xyz file file.xyz, and the geometry will be optimized at the SCF level. The loop will be terminated if the geometry converges, or maximum number of iterations (MaxIter) will be reached (the default value is 50).

There are several EMIL commands (see sect /refUG:sec:emil), which can be useful to control geometry optimization. For example, it is possible to execute some MOLCAS modules only once:

```
>> IF ( ITER = 1 )
* this part of the input will be executed only during the first iteration
>> ENDIF
```

4.2.22 Core and Embedding Potentials within the SEWARD Program

7.4 is able to perform *effective core potential* (ECP) and *embedded cluster* (EC) calculations. In ECP calculations [16, 17] the *core* electrons of a molecule are kept frozen and represented by a set of atomic effective potentials, while only the valence electrons are explicitly handled in the quantum mechanical calculation. In EC calculations only the electrons assigned to a piece of the whole system, the *cluster*, are explicitly treated in a quantum mechanical calculation, while the rest of the whole system, the *environment*, is kept frozen and represented by embedding potentials which act onto the *cluster*. For an explanation of the type of potentials and approaches used in MOLCAS the reader is referred to the section 8.36 of the user's guide.

To use such type of effective potentials implies to compute a set of atomic integrals and therefore involves only the `SEWARD` program. The remaining *MOLCAS* programs will simply use the integrals in the standard way and no indication of the use of ECP will appear in the outputs further on; the difference is of course that the absolute energies obtained for the different methods are not comparable to those obtained in an all-electron calculation. Therefore, the only input required to use ECP or EC is the `SEWARD` input, according to the examples given below. In the input files of the subsequent *MOLCAS* programs the orbitals corresponding to the excluded core orbitals should of course not be included, and not the excluded electrons.

`SEWARD` input for Effective Core Potential calculations

Astatine (At) is the atomic element number 85 which has the main configuration in its electronic ground state: [*core*] $6s^2 5d^{10} 6p^5$. In the *core* 68 electrons are included, corresponding to the xenon configuration plus the $4f^{14}$ lantanide shell. To perform an ECP calculation in a molecular system containing At it is necessary to specify which type of effective potential will substitute the *core* electrons and which valence basis set will complement it. Although the core ECP's (strictly AIMP's, see section 8.36 of the user's guide) can be safely mixed together with all-electron basis set, the valence basis sets included in the *MOLCAS* AIMP library have been explicitly optimized to complement the AIMP potentials.

The file *`AIMPLIB`* in the *MOLCAS* directory $MOLCAS/bin contains the list of available core potentials and valence basis sets. Both the relativistic (CG-AIMP's) and the nonrelativistic (NR-AIMP's) potentials are included. As an example, this is the head of the entry corresponding to the relativistic ECP for At:

```
/At.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.17e-CG-AIMP.
Z.Barandiaran, L.Seijo, J.Chem.Phys. 101(1994)4049; L.S. JCP 102(1995)8078.
core[Xe,4f] val[5d,6s,6p]  SO-corr  (11,1,1/9111/611*/4o1)=3s4p3d2f recommended
*
* - spin-orbit basis set correction from
*   L.Seijo, JCP 102(1995)8078.
*
* - (5o) f orthogonality function is the 4f core orbital
*
*ATQR-DSP(A3/A2/71/5)-SO       (A111/9111/611/41)
```

The first line is the label line written in the usual `SEWARD` format: element symbol, basis label, first author, size of the primitive set, size of the contracted set (in both cases referred to the valence basis set), and type of ECP used. In this case there are 17 valence electrons and the effective potential is a Cowan-Griffin-relativistic core AIMP. The number of primitive functions for the valence basis set ($13s12p8d5f$ here) will split into different subsets (within a segmented contraction scheme) according to the number of contracted functions. In the library, the contracted basis functions have been set to the minimal basis size: $1s1p2d1f$ for the valence electrons in At. This means the following partition: $1s$ contracted function including 13 primitive functions; $1p$ contracted function including 12 primitive functions; $2d$ contracted functions, the first one containing seven primitive functions and the second one primitive function (see the library), and finally $1f$ contracted function containing five primitive functions.

In the `SEWARD` input the user can modify the contraction scheme simply varying the number of contracted functions. There is a recommended size for the valence basis set which is printed in the third line for each atom entry on the library: $3s4p3d2f$ for At. For example, the simplest way to include the atom core potential and valence basis set in the `SEWARD` input would be:

```
At.ECP...3s4p3d2f.17e-CG-AIMP.␣/␣AIMPLIB
```

This means a partition for the valence basis set as showed in figure 4.17.

Figure 4.17: Partition of a valence basis set using the ECP's library

```
Basis set:AT.ECP...3S4P3D2F.17E-CG-AIMP.
                 Type
                  s
          No.      Exponent    Contraction Coefficients
           1   .133037396D+07  -.000154   .000000   .000000
           2   .993126141D+05  -.001030   .000000   .000000
           3   .128814005D+05  -.005278   .000000   .000000
           4   .247485916D+04  -.014124   .000000   .000000
           5   .214733934D+03   .069168   .000000   .000000
           6   .111579706D+03   .020375   .000000   .000000
           7   .370830653D+02  -.259246   .000000   .000000
           8   .113961072D+02   .055751   .000000   .000000
           9   .709430236D+01   .649870   .000000   .000000
          10   .448517638D+01  -.204733   .000000   .000000
          11   .157439587D+01  -.924035   .000000   .000000
          12   .276339384D+00   .000000  1.000000   .000000
          13   .108928284D+00   .000000   .000000  1.000000
                 Type
                  p
          No.      Exponent    Contraction Coefficients
          14   .608157825D+04   .000747   .000000   .000000   .000000
          15   .128559298D+04   .009304   .000000   .000000   .000000
          16   .377428675D+03   .026201   .000000   .000000   .000000
          17   .552551834D+02  -.087130   .000000   .000000   .000000
          18   .233740022D+02  -.044778   .000000   .000000   .000000
          19   .152762905D+02   .108761   .000000   .000000   .000000
          20   .838467359D+01   .167650   .000000   .000000   .000000
          21   .234820847D+01  -.290968   .000000   .000000   .000000
          22   .119926577D+01  -.237719   .000000   .000000   .000000
          23   .389521915D+00   .000000  1.000000   .000000   .000000
          24   .170352883D+00   .000000   .000000  1.000000   .000000
          25   .680660800D-01   .000000   .000000   .000000  1.000000
                 Type
                  d
          No.      Exponent    Contraction Coefficients
          26   .782389711D+03   .007926   .000000   .000000
          27   .225872717D+03   .048785   .000000   .000000
          28   .821302011D+02   .109617   .000000   .000000
          29   .173902999D+02  -.139021   .000000   .000000
          30   .104111329D+02  -.241043   .000000   .000000
          31   .195037661D+01   .646388   .000000   .000000
          32   .689437556D+00   .000000  1.000000   .000000
          33   .225000000D+00   .000000   .000000  1.000000
                 Type
                  f
          No.      Exponent    Contraction Coefficients
          34   .115100000D+03   .065463   .000000
          35   .383200000D+02   .270118   .000000
          36   .151600000D+02   .468472   .000000
          37   .622900000D+01   .387073   .000000
          38   .242100000D+01   .000000  1.000000
```

Therefore, the primitive set will always be split following the scheme: the first contracted function will contain the total number of primitives minus the number of remaining contracted functions and each of the remaining contracted functions will contain one single uncontracted primitive function. In the present example possible contraction patterns are: contracted $1s1p2d1f$ (13/12/8,1/5 primitives per contracted function, respectively), $2s2p3d2f$ (12,1/11,1/7,1,1/4,1), $3s3p4d2f$ (11,1,1/10,1,1/6,1,1,1/4,1), etc. Any other scheme which cannot be generated in this way must be included in the input using the Inline format for basis sets or an additional user's library. When the Inline option is used both the valence basis set and the AIMP potential must be included in the input, as it will be shown in the next section.

For an explanation of the remaining items in the library the reader is referred to the section 8.36 of the user's guide.

Figure 4.18 contains the sample input required to compute the SCF wave function for the astatine hydride molecule at an internuclear distance of 3.2 au. The Cowan-Griffin-relativistic core-AIMP has been used for the At atom with a size for the valence basis set recommended in the *AIMPLIB* library: $3s4p3d2f$.

Figure 4.18: Sample input required by SEWARD and SCF programs to compute the SCF wave function of HAt using a relativistic ECP

```
␣&GATEWAY
Title
HAt␣molecule␣using␣17e-Cowan-Griffin-relativistic␣core-AIMP
group
X␣Y
coord
HAt.xyz
Basis␣set
H.ano-l-vtzp
Basis␣set
At.ECP...3s4p3d2f.17e-CG-AIMP.␣/␣AIMPLIB
␣&SEWARD
␣&SCF
Title
␣HAt␣g.s.␣(At-val=5d,6s,6p)
Occupied
␣4␣2␣2␣1
```

`SEWARD` input for Embedded Cluster calculations

To perform embedded cluster (EC) calculations requires certain degree of experience and therefore the reader is referred to the literature quoted in section 8.36 of the user's guide. On the following a detailed example is however presented. It corresponds to EC calculations useful for local properties associated to a Tl^+ impurity in $KMgF_3$. First, a cluster must be specified. This is the piece of the system which is explicitly treated by the quantum mechanical calculation. In the present example the cluster will be formed by the unit $(TlF_{12})^{11-}$. A flexible basis for the cluster must be determined. Figure 4.20 contains the basis set selection for the thallium and fluorine atoms. In this case ECP-type basis sets have been selected. For Tl a valence basis set of size $3s4p4d2f$ has been used combined with the relativistic core-AIMP potentials as they appear in the *AIMPLIB* library. For the F atom the valence basis set has been modified from that appearing in the *AIMPLIB* library. In this case the exponent of the p-diffuse function and the p contraction coefficients of the

F basis set have been optimized in calculations on the fluorine anion included in the specific lattice in order to obtain a more flexible description of the anion. This basis set must be introduced Inline, and then also the ECP potential must be added to the input. The user can compare the basis set and ECP for F in figure 4.20 with the entry of *AIMPLIB* under /F.ECP.Huzinaga.5s6p1d.1s2p1d.7e-NR-AIMP. The entry for the Inline format must finish with the line End of Spectral Representation Operator.

Once the cluster has been defined it is necessary to represent the embedding lattice. Presently, *MOLCAS* includes embedding potentials for ions of several elpasolites, fluoro-perovskites, rocksalt structure oxides and halides, and fluorites. The embedding potentials for any other structure can be included in the input using the Inline format or included in a private user library. In the selected example a fluoro-perovskite lattice has been selected: $KMgF_3$. Here, the Tl^+ impurity substitutes a K^+ ion in an O_h site with 12 coordination. The first coordination shell of fluorine ions has been included into the cluster structure and the interactions to the Tl atom will be computed by quantum mechanical methods. The rest of the lattice will be represented by the structure $KMgF_3$ with five shells of ions at experimental sites. The shells have been divided in two types. Those shells closer to the cluster are included as embedding potentials from the library *EMP.AIMPLIB*. For example the potassium centers will use the entry on figure 4.19.

Figure 4.19: Sample input for an embedded core potential for a shell of potassium cations

```
Basis␣set
K.ECP..0s.0s.0e-AIMP-KMgF3.␣/␣EMB.AIMPLIB
PSEUdocharge
K2-1␣␣␣␣0.0000000000␣␣␣0.0000000000␣␣␣7.5078420000
K2-2␣␣␣␣0.0000000000␣␣␣7.5078420000␣␣␣0.0000000000
K2-3␣␣␣␣0.0000000000␣␣␣7.5078420000␣␣␣7.5078420000
K2-4␣␣␣␣7.5078420000␣␣␣0.0000000000␣␣␣0.0000000000
K2-5␣␣␣␣7.5078420000␣␣␣0.0000000000␣␣␣7.5078420000
K2-6␣␣␣␣7.5078420000␣␣␣7.5078420000␣␣␣0.0000000000
K2-7␣␣␣␣7.5078420000␣␣␣7.5078420000␣␣␣7.5078420000
End␣Of␣Basis
```

No basis set is employed to represent the potassium centers on figure 4.19, which just act as potentials embedding the cluster. The keyword PSEUDOCHARGE ensures that the interaction energy between the embedding potentials is not included in the "Nuclear repulsion energy" and that their location is not varied in a geometry optimization (`SLAPAF`). The first shells of Mg^{+2} and F^- will be introduced in the same way.

The remaining ions of the lattice will be treated as point charges. To add a point charge on the `SEWARD` input it is possible to proceed in two ways. One possibility is to employ the usual label to introduce an atom with its basis functions set to zero and the keyword CHARGE set to the value desired for the charge of the center. This way of introducing point charges must not be used when geometry optimizations with the `SLAPAF` program is going to be performed because `SLAPAF` will recognize the point charges as atoms whose positions should be optimized. Instead the keyword XFIELD can be used as it is illustrated in figure 4.20. XFIELD must be followed by a line containing the number of point charges, and by subsequent lines containing the cartesian coordinates and the introduced charge or the three components of the dipole moment at the specified geometry. In any case the seven positions in each line must be fulfilled. To ensure the neutral character of the whole system the point charges placed on the terminal edges, corners or faces of the lattice must have the proper fractional values.

Figure 4.20 contains the complete sample input to perform a SCF energy calculation on the system $(TlF_{12})^{11-} : KMgF_3$.

Figure 4.20: Sample input for a SCF geometry optimization of the $(TlF_{12})^{11-} : KMgF_3$ system

```
␣&SEWARD␣&END
Title
|␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣Test␣run␣TlF12:KMgF3.1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
|**␣Molecule␣**␣␣␣(TlF12)11-␣cluster␣embedded␣in␣a␣lattice␣of␣KMgF3␣␣␣␣␣␣␣␣␣␣␣␣|
|**␣Basis␣set␣and␣ECP␣**␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
|␣␣*␣Tl␣*␣(11,1,1/9,1,1,1/5,1,1,1/4,1)␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣from␣AIMPLIB|
|␣␣␣␣␣␣␣␣␣13e-Cowan-Griffin-relativistic␣core-AIMP␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣from␣AIMPLIB|
|␣␣*␣F␣*␣␣(4,1/4,1,1)␣diffuse-p␣optimized␣in␣KMgF3:F(-)␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣inline|
|␣␣␣␣␣␣␣␣␣␣7e-nonrelativistic␣core-AIMP␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣inline|
|␣␣KMgF3␣embedding-AIMPs␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣from␣EMB.AIMPLIB|
|**␣cluster␣geometry␣**␣␣␣r(Tl-F)/b=␣5.444␣=␣3.84948932␣*␣sqrt(2)␣␣␣␣␣␣␣␣␣␣␣␣␣␣|
|**␣lattice␣**␣␣(perovskite␣structure)␣5␣shells␣of␣ions␣at␣experimental␣sites␣␣|
Symmetry
X␣Y␣Z
Basis␣set
Tl.ECP.Barandiaran.13s12p8d5f.3s4p4d2f.13e-CG-AIMP.␣/␣AIMPLIB
Tl␣␣␣␣␣0.00000␣␣␣0.00000␣␣␣0.00000
End␣Of␣Basis
Basis␣set
F.ECP....␣/␣Inline
*␣␣␣␣basis␣set␣and␣core-AIMP␣as␣in:␣F.ECP.Huzinaga.5s6p1d.2s4p1d.7e-NR-AIMP.
*␣␣␣␣except␣that␣the␣p-diffuse␣and␣the␣p␣contraction␣coeffs.␣have␣been
*␣␣␣␣optimized␣in␣KMgF3-embedded␣F(-)␣scf␣calculations.
␣␣7.000000␣␣␣␣␣␣␣␣␣1
␣␣␣␣5␣␣␣␣2
␣␣␣405.4771610
␣␣␣61.23686380
␣␣␣13.47117730
␣␣␣1.095173720
␣␣␣.3400847530
␣␣-.013805187800␣␣␣.000000000000
␣␣-.089245064800␣␣␣.000000000000
␣␣-.247937861000␣␣␣.000000000000
␣␣␣.632895340000␣␣␣.000000000000
␣␣␣.000000000000␣␣␣.465026336000
␣␣␣␣6␣␣␣␣3
␣␣␣44.13600920
␣␣␣9.982597110
␣␣␣2.947082680
␣␣␣.9185111850
␣␣␣.2685213550
␣␣␣.142
␣␣␣.015323038700␣␣␣.000000000000␣␣␣.000000000000
␣␣␣.095384703000␣␣␣.000000000000␣␣␣.000000000000
␣␣␣.291214218000␣␣␣.000000000000␣␣␣.000000000000
␣␣␣.441351868000␣␣␣.000000000000␣␣␣.000000000000
␣␣␣.000000000000␣␣␣.427012588000␣␣␣.000000000000
␣␣␣.000000000000␣␣␣.000000000000␣␣1.000000000000
*
*␣Core␣AIMP:␣F-1S
*
*␣Local␣Potential␣Paramenters␣:␣(ECP␣convention)
*␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣A(AIMP)=-Zeff*A(ECP)
M1
␣␣␣␣7
␣␣␣279347.4000
␣␣␣31889.74900
␣␣␣5649.977600
```

```
␣␣␣1169.273000
␣␣␣269.0513200
␣␣␣71.29884600
␣␣␣22.12150700
␣␣␣.004654725000
␣␣␣.007196816857
␣␣␣.015371258571
␣␣␣.032771900000
␣␣␣.070383742857
␣␣␣.108683807143
␣␣␣.046652035714
M2
␣␣␣␣0
COREREP
␣␣␣1.0
PROJOP
␣␣␣␣0
␣␣␣14␣␣␣␣1
␣␣52.7654040
␣␣␣210965.4100
␣␣␣31872.59200
␣␣␣7315.837400
␣␣␣2077.215300
␣␣␣669.9991000
␣␣␣232.1363900
␣␣␣84.99573000
␣␣␣32.90124100
␣␣␣13.36331800
␣␣␣5.588141500
␣␣␣2.319058700
␣␣␣.9500928100
␣␣␣.3825419200
␣␣␣.1478404000
␣␣␣.000025861368
␣␣␣.000198149380
␣␣␣.001031418900
␣␣␣.004341016600
␣␣␣.016073698000
␣␣␣.053856655000
␣␣␣.151324390000
␣␣␣.318558040000
␣␣␣.404070310000
␣␣␣.190635320000
␣␣␣.011728993000
␣␣␣.002954046500
␣␣-.000536098280
␣␣␣.000278474090
*
Spectral␣Representation␣Operator
Valence␣primitive␣basis
Exchange
End␣of␣Spectral␣Representation␣Operator
F_1␣␣␣␣␣␣␣␣3.849489320␣␣␣␣␣␣␣3.849489320␣␣␣␣␣␣␣␣.000000000
F_2␣␣␣␣␣␣␣␣␣.000000000␣␣␣␣␣␣␣3.849489320␣␣␣␣␣␣␣3.849489320
F_3␣␣␣␣␣␣␣␣3.849489320␣␣␣␣␣␣␣␣.000000000␣␣␣␣␣␣␣3.849489320
*␣3*4␣=␣12
End␣Of␣Basis
*␣end␣of␣cluster␣data:␣TlF12
*␣beginning␣of␣lattice␣embedding␣data:␣KMgF3
Basis␣set
K.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.␣/␣EMB.AIMPLIB
pseudocharge
*␣K(+)␣ions␣as␣embedding␣AIMPs
K2-1␣␣␣␣0.0000000000␣␣␣0.0000000000␣␣␣7.5078420000
K2-2␣␣␣␣0.0000000000␣␣␣7.5078420000␣␣␣0.0000000000
K2-3␣␣␣␣0.0000000000␣␣␣7.5078420000␣␣␣7.5078420000
```

```
K2-4␣␣␣␣7.5078420000␣␣␣0.0000000000␣␣␣0.0000000000
K2-5␣␣␣␣7.5078420000␣␣␣0.0000000000␣␣␣7.5078420000
K2-6␣␣␣␣7.5078420000␣␣␣7.5078420000␣␣␣0.0000000000
K2-7␣␣␣␣7.5078420000␣␣␣7.5078420000␣␣␣7.5078420000
*␣3*2␣+␣3*4␣+␣1*8␣=␣26
End␣Of␣Basis
Basis␣set
Mg.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.␣/␣EMB.AIMPLIB
pseudocharge
*␣Mg(2+)␣ions␣as␣embedding␣AIMPs
MG1-1␣␣␣3.7539210000␣␣␣3.7539210000␣␣␣3.7539210000
MG3-1␣␣␣3.7539210000␣␣␣3.7539210000␣␣11.2617630000
MG3-2␣␣␣3.7539210000␣␣11.2617630000␣␣␣3.7539210000
MG3-3␣␣␣3.7539210000␣␣11.2617630000␣␣11.2617630000
MG3-4␣␣11.2617630000␣␣␣3.7539210000␣␣␣3.7539210000
MG3-5␣␣11.2617630000␣␣␣3.7539210000␣␣11.2617630000
MG3-6␣␣11.2617630000␣␣11.2617630000␣␣␣3.7539210000
MG3-7␣␣11.2617630000␣␣11.2617630000␣␣11.2617630000
*␣8*8␣=␣64
End␣Of␣Basis
Basis␣set
F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.␣/␣EMB.AIMPLIB
pseudocharge
*␣F(-)␣ions␣as␣embedding␣AIMPs
F2-1␣␣␣␣3.7539210000␣␣␣3.7539210000␣␣␣7.5078420000
F2-2␣␣␣␣3.7539210000␣␣␣7.5078420000␣␣␣3.7539210000
F2-3␣␣␣␣7.5078420000␣␣␣3.7539210000␣␣␣3.7539210000
F3-1␣␣␣␣0.0000000000␣␣␣3.7539210000␣␣11.2617630000
F3-2␣␣␣␣3.7539210000␣␣␣0.0000000000␣␣11.2617630000
F3-3␣␣␣␣3.7539210000␣␣11.2617630000␣␣␣0.0000000000
F3-4␣␣␣␣0.0000000000␣␣11.2617630000␣␣␣3.7539210000
F3-5␣␣␣␣3.7539210000␣␣11.2617630000␣␣␣7.5078420000
F3-6␣␣␣␣0.0000000000␣␣11.2617630000␣␣11.2617630000
F3-7␣␣␣␣3.7539210000␣␣␣7.5078420000␣␣11.2617630000
F3-8␣␣␣11.2617630000␣␣␣3.7539210000␣␣␣0.0000000000
F3-9␣␣␣11.2617630000␣␣␣0.0000000000␣␣␣3.7539210000
F3-10␣␣␣11.2617630000␣␣␣3.7539210000␣␣␣7.5078420000
F3-11␣␣␣␣7.5078420000␣␣␣3.7539210000␣␣11.2617630000
F3-12␣␣␣11.2617630000␣␣␣0.0000000000␣␣11.2617630000
F3-13␣␣␣11.2617630000␣␣11.2617630000␣␣␣0.0000000000
F3-14␣␣␣␣7.5078420000␣␣11.2617630000␣␣␣3.7539210000
F3-15␣␣␣11.2617630000␣␣␣7.5078420000␣␣␣3.7539210000
F3-16␣␣␣11.2617630000␣␣11.2617630000␣␣␣7.5078420000
F3-17␣␣␣␣7.5078420000␣␣11.2617630000␣␣11.2617630000
F3-18␣␣␣11.2617630000␣␣␣7.5078420000␣␣11.2617630000
*␣9*4␣+␣␣12*8␣=␣132
End␣Of␣Basis
*␣The␣rest␣of␣the␣embedding␣lattice␣will␣be␣represented␣by␣point␣charges,
*␣which␣enter␣into␣the␣calculation␣in␣the␣form␣of␣a␣XField.
*
XField
␣95
*
*␣K(+)␣ions␣as␣point␣charges
␣␣␣␣0.0000000000␣␣␣0.0000000000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣0.0000000000␣␣␣7.5078420000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣0.0000000000␣␣15.0156840000␣␣␣0.0000000000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣0.0000000000␣␣15.0156840000␣␣␣7.5078420000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣0.0000000000␣␣15.0156840000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣7.5078420000␣␣␣0.0000000000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣7.5078420000␣␣␣7.5078420000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣7.5078420000␣␣15.0156840000␣␣␣0.0000000000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣7.5078420000␣␣15.0156840000␣␣␣7.5078420000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣␣7.5078420000␣␣15.0156840000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣0.0000000000␣␣␣0.0000000000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣0.0000000000␣␣␣7.5078420000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
```

```
␣␣␣15.0156840000␣␣␣0.0000000000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣7.5078420000␣␣␣0.0000000000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣7.5078420000␣␣␣7.5078420000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣7.5078420000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣15.0156840000␣␣␣0.0000000000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣15.0156840000␣␣␣7.5078420000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣15.0156840000␣␣15.0156840000␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
*
*␣F(-)␣ions␣as␣point␣charges
␣␣␣3.7539210000␣␣␣3.7539210000␣␣15.0156840000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣11.2617630000␣␣15.0156840000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣15.0156840000␣␣␣3.7539210000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣15.0156840000␣␣11.2617630000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣␣3.7539210000␣␣15.0156840000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣11.2617630000␣␣15.0156840000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣15.0156840000␣␣␣3.7539210000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣15.0156840000␣␣11.2617630000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣15.0156840000␣␣␣3.7539210000␣␣␣3.7539210000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣␣3.7539210000␣␣11.2617630000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣11.2617630000␣␣␣3.7539210000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
␣␣␣15.0156840000␣␣11.2617630000␣␣11.2617630000␣␣␣␣␣␣␣-1.0␣␣0.␣␣0.␣␣0.
*
*␣Mg(2+)␣ions␣in␣face,␣as␣fractional␣point␣charges
␣␣␣3.7539210000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣␣3.7539210000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣␣3.7539210000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣11.2617630000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣11.2617630000␣␣␣3.7539210000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣11.2617630000␣␣11.2617630000␣␣␣␣␣␣␣␣+1.0␣␣0.␣␣0.␣␣0.
*
*␣Mg(2+)␣ions␣in␣edge,␣as␣fractional␣point␣charges
␣␣␣3.7539210000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
␣␣11.2617630000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
␣␣18.7696050000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣+0.5␣␣0.␣␣0.␣␣0.
*
*␣Mg(2+)␣ions␣in␣corner,␣as␣fractional␣point␣charges
␣␣18.7696050000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣␣+0.25␣␣0.␣0.␣0.
*
*␣F(-)␣ions␣in␣face,␣as␣fractional␣point␣charges
␣␣␣0.0000000000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣␣0.0000000000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣0.0000000000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣␣7.5078420000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣18.7696050000␣␣␣0.0000000000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣0.0000000000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣18.7696050000␣␣␣7.5078420000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣0.0000000000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣18.7696050000␣␣15.0156840000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣3.7539210000␣␣15.0156840000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣7.5078420000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣11.2617630000␣␣␣0.0000000000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣7.5078420000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣11.2617630000␣␣␣7.5078420000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣11.2617630000␣␣18.7696050000␣␣␣0.0000000000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣7.5078420000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣11.2617630000␣␣18.7696050000␣␣␣7.5078420000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣␣7.5078420000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
```

```
␣␣11.2617630000␣␣18.7696050000␣␣15.0156840000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣11.2617630000␣␣15.0156840000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣␣0.0000000000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣0.0000000000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣␣7.5078420000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣0.0000000000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣3.7539210000␣␣15.0156840000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣15.0156840000␣␣␣3.7539210000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣11.2617630000␣␣␣0.0000000000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣7.5078420000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣11.2617630000␣␣␣7.5078420000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣7.5078420000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣11.2617630000␣␣15.0156840000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣15.0156840000␣␣11.2617630000␣␣18.7696050000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣15.0156840000␣␣18.7696050000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣15.0156840000␣␣␣3.7539210000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣15.0156840000␣␣18.7696050000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
␣␣18.7696050000␣␣15.0156840000␣␣11.2617630000␣␣␣␣␣␣␣-0.5␣␣0.␣0.␣0.
*
*␣F(-)␣ions␣in␣edge,␣as␣fractional␣point␣charges
␣␣␣0.0000000000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣␣7.5078420000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣0.0000000000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣␣7.5078420000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣18.7696050000␣␣␣0.0000000000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣18.7696050000␣␣␣7.5078420000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣18.7696050000␣␣15.0156840000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣15.0156840000␣␣18.7696050000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
␣␣18.7696050000␣␣15.0156840000␣␣18.7696050000␣␣␣␣␣␣␣-0.25␣␣0.␣0.␣0.
*␣␣end␣of␣lattice␣embedding␣data:␣KMgF3
*␣13␣cluster␣components␣␣and␣881␣lattice␣components
End␣of␣input
␣&SCF␣&END
Title
␣(TlF12)11-␣run␣as␣D2h
Occupied
␣12␣␣␣␣7␣␣␣␣7␣␣␣␣6␣␣␣␣7␣␣␣␣6␣␣␣␣6␣␣␣␣3
End␣of␣input
```

4.2.23 GRID_IT: A Program for Orbital Visualization

`GRID_IT` is an interface program for calculations of molecular orbitals and density in a set of cartesian grid points. Calculated grid can be visualized by separate program (`MOLCAS GV`) in the form of isosurfaces.

`GRID_IT` generates the regular grid and calculates amplitudes of molecular orbitals in this net. Keywords SPARSE,DENSE, NPOINTS specify the density of the grid. And keywords ORANGE (occupation range), ERANGE (energy range), SELECT allow to select some specific orbitals to draw.

As default `GRID_IT` will use grid net with intermediate quality, and choose orbitals near HOMO-LUMO region. Note, that using keyword ALL - to calculate grids for all orbitals or DENSE - to calculate grid with very high quality you can produce a very huge output file.

`GRID_IT` requires the communication file *`RUNFILE`*, processed by `GATEWAY` and any formated *`INPORB`* file: *`SCFORB`*, *`RASORB`*, *`PT2ORB`*, generated by program `SCF`, `RASSCF`, or `CASPT2`, respectively. The output file *`M2MSI`* contains the graphical information.

Normally you do not need to specify any keywords for `GRID_IT`: the selection of grid size, as well as the selection of orbitals done automatically.

An input example for `GRID_IT` is:

```
␣&GRID_IT
Dense
*␣compute␣orbitals␣from␣20␣to␣23␣form␣symmetry␣1␣and␣orbital␣4␣from␣symmetry␣2
SELECT
1:20-23,2:4
```

`GRID_IT` can be run in a sequence of other computational codes (if you need to run `GRID_IT` several times, you have to rename grid file by using EMIL command, or by using keyword NAME)

```
&GATEWAY
␣...
&SEWARD
&SCF
&GRID_IT
NAME
scf
&RASSCF
&GRID_IT
NAME
ras
```

or, you can run `GRID_IT` separately, when a calculation finished.

```
&GATEWAY
␣...
>>LINK␣/home/joe/project/water/water.ScfOrb␣INPORB
&GRID_IT
or,
&GATEWAY
␣...
&GRID_IT
FILEORB
/home/joe/project/water/water.ScfOrb
```

This is quite important to understand that the timing for `GRID_IT`, and the size of generated grid file depends dramatically on the targeting problem. To get a printer quality pictures you have to use Dense grid, but in order to see the shape of orbitals - low quality grids are much more preferable.

The following table illustrates this dependence:

C_{24} molecule, 14 orbitals.

Keywords	Time (sek)	filesize	picture quality
Dense, ASCII	188	473 Mb	best
Dense	117	328 Mb	best
Dense, Pack	117	41 Mb	below average
Default (no keywords)	3	9 Mb	average
Pack	3	1.4 Mb	average
Sparse	1.3	3 Mb	poor
Sparse, Pack	1.3	620 Kb	poor

4.3 MOLCAS grid and geometry viewer

4.3.1 Editing molecular structures using the `GV` program.

Program `GV` is used to manipulate molecular structures, build new molecules, etc. It is also used to show orbitals , densities, etc. Here, we shall exemplify how structures are manipulated. The program requires some coordinates to start with. One possible source of initial coordinates is *Coord* directory, which contains a large number of molecular structures. Change directory to *Coord* under the main directory of *MOLCAS*, and run for example molcas gv Water.xyz.

In order to make modifications of coordinates we have to select one, two, three or four atoms. Selection is made by clicking on an atom. The first selected atom is covered by blue-colored net, the following selected atoms are covered by magenta-colored net. Any time the user can press 'space' key, or click by middle button on the mouse to remove selection. If only one atom is selected - you can make modifications for this atom, if two atoms are selected - you can modify the bond between atoms, if three atoms are selected you can modify the angle, and for four atoms the dihedral angle. Note, that modifying bond or angle - the first selected atoms will move. Modifications can be done by the following keyboard shortcuts: '+/-', 'PageUp/PageDown',F4.

Pressing + or - will modify the value, e.g. if a bond is selected, and user pressed '+' key, the bond length will increase, so the first selected atom will move away from the second atom.

Pressing PageUp/PageDown will modify the 'property' of selected object. If only one atom is selected - it will change element name, if bond is selected - it will change the type of the bond (single, double, etc.), and for angle selected, it will change the angle to most common values.

If F4 (or '=') key is pressed, user can type the value in a separate input box. If one atom is selected, user can type an element name. If a bond is selected, user can type the lenght of the bond, and finally if an angle is selected - user can type the new value of the angle.

Any time a Backspace key can be used to revert modifications (Undo).

F2 key can be used to save the coordinate file. The file name will be generated from the original name, by adding a counter. Shift-F2 will overwrite the original file.

If an atom is selected it is possible to use Delete or Insert key to delete or insert a new atom. Let make H_2O_2 molecule started from Water.xyz file.

- select a H atom
- press PageUp until the atom becomes oxygen.
- select another O atom
- press F4 and type 1.474 followed by Enter
- press Space bar to un select atoms.
- select H-O-O
- press F4 and type 94.8 followed by Enter

- press Space button
- select 'new' Oxygen atom
- press Insert to add H atom near selected atom
- select two oxygen atoms
- press F4 and type 94.8
- select remaining hydrogen atom to select dihedral angle
- press F4 and type 111.5
- fix O-H bond lenghts.

Now we will continue to edit the H_2O_2 molecule. If we select the O-O bond and change interatomic distance - only one atom will move. If we want to move a group (O-H), we have to mark this group first. There are two different ways to mark atoms into a group. One can do this manually, clicking on an atom with pressed Shift button. Or, it is possible to mark connected (bonded) atoms, in respect to a selected bond. Select an O-O bond and press F7 button. All atoms, which are connected to the first selected atom will be 'marked', and shown as cyan-colored. If you change the length of the O-O bond, all marked group will move accordigly. Note that unselection (Space key, or mouse middle click) will remove selection first, and the second use of unselect button will remove marking of a group.

The modified value (of a bond lenght or an angle) is shown on the information line of the screen. Sometime you would like to observe another value during a modification of coordinates. To achive such behavour, select a bond, or an angle, and press F6 button. Now you can make another selection and make modifications in the geometry. But in this case originally selected value will be watched. Pressing Shift-F6 key switches off the watching mode.

`GV` contains a short list of molecular fragments, which can be added to a molecule. Press F3 key, to get a list of available fragmets. Clicking on a picture with a fragment, you will add this fragment into your sceene. If no atoms are selected, the fragment will be added somewhere around the current molecule. If one atom is selected, the fragment will be inserted near the selected atom. Note, that once a fragment has been selected, an Insert key will insert this fragment.

Example. Let's make mesitylene (1,3,5-trimethylbenzene).

- molcas gv Benzene
- mark (not select) 3 hydrogen atoms
- Press Delete
- select C atom, press F3 and click on CH3 icon
- select another C atom, and press Insert
- select third C atom, and press Insert

To make modifications of coordinates via distances and angles, you might need dummy (reference) atoms. These dummy atoms can be set by End button. If there is no selected atoms, 'End' key will add dummy atoms located on XYZ axeses. If a bond is selected, the dummy atom will be placed in the middle of the bond. For example, if you have a planar molecule, but it is not oriented according to cartesian axeses, you can add dummy atoms on each axis, mark all atoms in the molecule, and select a dihedral angle between the plain of the molecule and desired plane, created by dummy atoms.

Program `GV` can recognize the symmetry elements of a molecule, or apply symmetry operations for all, or marked atoms in the molecule. If no selection is made, F8 key displays symmetry elements of the molecule (D_{2h} subgroup only). If some atoms are selected, F8 key will apply a symmetry operation: in case of only one atoms being selected inversion, in case of a bond, C_2 axis, and in case of an angle - a mirror plain. This feature can be used to construct symmetrical molecules. Let us start from a benzene molecule. Delete two hydrogens (in ortho positions), and select two carbon atoms (with broken bonds). Pressing F8 key will duplicate the structure, creating naphtalene.

If, during such transformation coordinates of a new atom are very close to another atom - the average coordinates will be used. For example, if you have an almost planar molecule, you can flatten it, by applying a mirror plane symmetry. Also note, that if a part of the molecule is marked, the symmetry operation will be applied only for marked part.

`GV` can be customized. Press F9 key to save current setting. It will create a directory *.molcasgv* in user HOME directory, with default settings for used colors, initial sizes etc. User can edit this files to reset the default values.

4.3.2 Visualization of orbitals with gv.

To visulaize orbitals and density by program `GV` you have to compute a grid file (*.grid*) first by using `GRID_IT`.

Orbitals can be browsed by PageUp/PageDown key, or selected by a menu, invoked by the right mouse button. If you know the symmetry and number of an orbital, you would like to display, you can press F4 (or =) key, and press # followed by symmetry and orbital number, e.g. (#13).

In order to change isosurface value, you can use + or - key, or press F4 key, and type a desired isosurface value.

Sometime you would like to filter orbitals, shown by `GV`. Pressing Delete key you can hide an orbital. All hidden orbitals will become visible if Insert key is pressed. Alternatevely, you can apply a filter to hide some orbitals by a criteria: symmetry number (s), orbital energy (e), occupation number (o), or typeindex (i). Usage of filters is clear from the following example: Press F4 key and type $\# :$ followed by a filter command - $\# : s14$ to display orbitals only from symmetry 1 and 4, $\# : e-2:1$ to display orbitals in an energy range between -2 and 1.

When the grid file is loaded, `GV` displays subspaces (frozen, inactive, RAS1, RAS2, RAS3, secondary, deleted). User can modify the typeindex of the orbital, save (F2 key) the INPORB file (it will have an extension *GvOrb*), and use this file in the following `RASSCF` calculation withou having to reorder the orbitals. In order to modify the index of the displayed molecule, user can use a menu, or press one of the keys: fi123sd. Pressing Space key (or middle mouse button) changes the typeindex in a loop.

It is possible to display all orbitals of the grid file simultaneously. Press F3 key to get the screen with all orbitals. By default, the background (rainbow colors) for each orbital corresponds to the type index information. Clicking on an individual orbital you can use the same keys to modify it's type, or delete it from the screen. Pressing F3 button again, or Escape will close the multiview mode. Using PageUp/PageDown in multiview mode will increase/decrease the sizes of subscreens. These features of `GV` can be quite helpful for selecting the different orbital spaces in RASSCF calculations.

`GV` can also be used to compare densities from different `GRID_IT` calculations. A command molcas gv -a 1.0 scf.grid rasscf.grid will compute a density difference between two grid file. A bit more complicated procedure should be done if you would like to visualize a density difference between two interactive molecules. First, you have to make a BSSE-like calculation, adding dummy atoms (with or without basis) in order to produce identical (in space grids). Let say, we computed 3 grid files *`A.grid`*, *`B.grid`*, *`AB.grid`*. Run molcas gv -a 1.0 A.grid B.grid –out sum.grid. `GV` will create a file *`sum.grid`* which is a sum (due to -a 1.0) of densities. Now, run molcas gv -a -1.0 AB.grid sum.grid to compute the difference AB - (A+B).

4.3.3 Writing MOLDEN input

By default the `SCF`, `RASSCF`, `CASVB`, `SLAPAF`, and `MCLR` modules generate input in Molden format. The `SCF` and `RASSCF` modules generate input for molecular orbital analysis, `CASVB` for valence bond orbital analysis, `SLAPAF` for geometry optimization analysis, and the `MCLR` module generates input for analysis of harmonic frequencies. Molden files can be visualized by `GV` or by `MOLDEN` (http://www.caos.kun.nl/˜schaft/molden/molden.html).

The generic name of the input file is MOLDEN. However, the actual name is different for the nodes as a reflection on the data generated by each module. Hence, the actual names for MOLDEN in each module are

- SCF module: $Project.scf.molden
- RASSCF module: $Project.rasscf.molden for the state-averaged natural orbitals, and $Project.rasscf.x.molden for the state-specific natural spin orbitals, where x is the index of a CI root.
- CASVB module: $Project.casvb.molden
- SLAPAF module: $Project.geo.molden
- MCLR module: $Project.freq.molden

4.4 Most frequent error messages found in MOLCAS

Due to the large number of systems where the *MOLCAS* package is executed and the large number of options included in each of the programs it is not possible to compile here all the possible sources of errors and error messages occurring in the calculations. The *MOLCAS* codes contain specific error message data basis where the source of the error and the possible solution is suggested. Unfortunately it is almost impossible to cover all the possibilities. Here the user will find a compendium of the more usual errors showing up in MOLCAS and the corresponding error messages.

Many of the error messages the user is going to obtain are specific for the operative system or architecture being used. The most serious ones are in most of cases related with compiler problems, operative system incompatibilities, etc. Therefore the meaning of this errors must be checked in the proper manuals or with the computer experts, and if they are characteristic only of MOLCAS, with MOLCAS authors. The most common, however, are simple mistakes related to lack of execution or reading permission of the shell scripts, MOLCAS executable modules, etc.

In the following the most usual errors found in MOLCAS are listed.

- The shell is unable to find the command molcas. The message in this case is, for instance:

  ```
  molcas:  not found
  ```

  The solution is to add into the PATH the location of molcas driver script.

- If the MOLCAS environment is not properly installed the first message showing up in the default error file is:

  ```
  ***
  *** Error: Could not find molcas driver shell
  *** Currently MOLCAS=
  ```

  Typing a command molcas, you can check which molcas installation will be used. Check the value of the variable MOLCAS, and define it in order to point to the proper location of molcas installation.

- Environment is not defined

  An attempt to run an executable without molcas driver scripts gives an error:

  ```
  Usage: molcas module_name input
  ```

- A call for a program can find problems like the three following ones:

  ```
  Program NNNN is not defined
  ```

  An error means that requested module is missing or the package is not installed.

- When the input file required for a MOLCAS program is not available, the program will not start at all and no output will be printed, except in the default error file where the following error message will appear:

  ```
  Input file specified for run subcommand not found : seward
  ```

- All the codes communicate via file *RUNFILE*, if for a some reason the file is missing or corrupted, you will get an error

  ```
  ***    Record not found in runfile
  ```

  The simple solution - restart seward to generate proper *RUNFILE*

- All the codes need integral files generated by `SEWARD` in files *`ONEINT`* and *`ORDINT`*. Even the direct codes need the one-electron integrals stored in *`ONEINT`*. The most common problem is then that a program fails to read one of this files because `SEWARD` has not been executed or because the files are read in the wrong address. Some of the error messages found in those cases are listed here.

  In the `SCF` module, the first message will appear when the one-electron integral file is missing and the second when the two-electron integral file is missing:

```
Two-electron integral file was not found!
 Try keyword DIRECT in SEWARD.
```

- *MOLCAS* use dynamical allocation of memory for temporary arrays. An error message 'Insufficient memory' means that requested value is too small - you have to specify MOLCASMEM variable and restart your calculation.

- if user ask to allocate (via MOLCASMEM) an amount of memory, which is large than possible on this computer, the following error message will be printed.

```
MA error: MA_init: could not allocate 2097152152 bytes
The initialization of the memory manager failed ( iRc=  1 ).
```

- An improper input (e.g. the code expects to read more numbers, than user specified in input file) will terminate the code with errorcode 20. If `AUTO` was used to parse input file, the code will locate the place in the input file, where the error occurred.

- Input/Output (I/O) problems are common, normally due to insufficient disk space to store the two-electron integral files or some of the intermediate files used by the programs. The error message would depend on the operative system used. An example for the `SCF` is shown below:

```
*******************************************************************************
 *******************************************************************************
 ***                                                                         ***
 ***                                                                         ***
 ***    Location: AixRd                                                      ***
 ***    File: ORDINT                                                         ***
 ***                                                                         ***
 ***                                                                         ***
 ***    Premature abort while reading buffer from disk:                      ***
 ***    Condition: rc != LenBuf                                              ***
 ***    Actual   :                0!=          262144                        ***
 ***                                                                         ***
 ***                                                                         ***
 *******************************************************************************
 *******************************************************************************
```

  The error indicates that the file is corrupted, or there is a bug in the code.

Section 5

Examples

5.1 Computing high symmetry molecules.

MOLCAS makes intensive use of the symmetry properties of the molecular systems in all parts of the calculation. The highest symmetry point group available, however, is the D_{2h} point group, which makes things somewhat more complicated when the molecule has higher symmetry. One of such cases is the calculation of linear molecules. In this section we describe calculations on different electronic states of three diatomic molecules: NiH, a heteronuclear molecule which belongs to the $C_{\infty v}$ symmetry group and C_2 and Ni_2, two homonuclear molecules which belong to the $D_{\infty h}$ symmetry group. They must be computed in *MOLCAS* using the lower order symmetry groups C_{2v} and D_{2h}, respectively, and therefore some codes such RASSCF use specific tools to constrain the resulting wave functions to have the higher symmetry of the actual point group. It must be pointed out clearly that linear symmetry cannot always be fully obtained in *MOLCAS* because the tools to average over degenerate representations are not totally implemented presently in the RASSCF program. This is the case, for instance, for the δ orbitals in a C_{2v}-$C_{\infty v}$ situation, as will be shown below. (For problems related to accurate calculations of diatomic molecules and symmetry see Ref. [18] and [19], respectively.). In a final section we will briefly comment the situation of high symmetry systems other than linear.

5.1.1 A diatomic heteronuclear molecule: NiH

Chemical bonds involving transition-metal atoms are often complex in nature due to the common presence of several unpaired electrons resulting in many close-lying spectroscopic states and a number of different factors such spin-orbit coupling or the importance of relativistic effects. NiH was the first system containing a transition-metal atom to be studied with the CASSCF method [20]. The large dynamic correlation effects inherent in a $3d$ semi-occupied shell with many electrons is a most severe problem, which few methods have been able to compute. The calculated dipole moment of the system has become one measurement of the quality of many *ab initio* methods [21]. We are not going to analyze the effects in detail. Let us only say that an accurate treatment of the correlation effects requires high quality methods such as MRCI, ACPF or CASPT2, large basis sets, and an appropriate treatment of relativistic effects, basis set superposition errors, and core-valence correlation. A detailed CASPT2 calculation of the ground state of NiH can be found elsewhere [22].

The ^3F ($3d^84s^2$) and ^3D ($3d^94s^1$) states of the nickel atom are almost degenerate with a splitting of only 0.03 eV [23] and are characterized by quite different chemical behavior. In systems such as the $^2\Delta$ ground state of NiH molecule, where both states take part in the bonding, an accurate description of the low-lying Ni atomic states is required. The selection of the active space for NiH is not trivial. The smallest set of active orbitals for the $^2\Delta$ ground state which allows a proper dissociation and also takes into account the important $3d\sigma$ correlation comprises the singly occupied $3d_{xy}$ orbital and three σ orbitals ($3d_{z^2}, \sigma$, and σ^*). One cannot however expect to obtain accurate enough molecular properties just by including non-dynamical correlation effects. MRCI+Q calculations with the most important CASSCF configurations in the reference space proved that at least one additional $3d\delta$ ($3d_{x^2-y^2}$) and its correlating orbital were necessary to obtain spectroscopic constants in close agreement with the experimental values. It is, however, a larger active space comprising all the eleven valence electrons distributed in twelve active orbitals (σ, σ^*, d, d') that is the most consistent choice of active orbitals as evidenced in the calculation of other metal hydrides such as CuH [22] and in the electronic spectrum of the Ni atom [23]. This is the active space we are going to use in the following example. We will use the ANO-type basis set contracted to Ni [$5s4p3d1f$] / H [$3s2p$] for simplicity. In actual calculations g functions on the transition metal and d functions on the hydrogen atom are required to obtain accurate results.

First we need to know the behavior of each one of the basis functions within each one of the symmetries. Considering the molecule placed in the z axis the classification of the spherical harmonics into the $C_{\infty v}$ point group is:

Table 5.1: Classification of the spherical harmonics in the $C_{\infty v}$ group.

Symmetry	Spherical harmonics					
σ	s	p_z	d_{z^2}	f_{z^3}		
π	p_x	p_y	d_{xz}	d_{yz}	$f_{x(z^2-y^2)}$	$f_{y(z^2-x^2)}$
δ	$d_{x^2-y^2}$	d_{xy}	f_{xyz}	$f_{z(x^2-y^2)}$		
ϕ	f_{x^3}	f_{y^3}				

In C_{2v}, however, the functions are distributed into the four representations of the group and therefore different symmetry representations can be mixed. The next table lists the distribution of the functions in C_{2v} and the symmetry of the corresponding orbitals in $C_{\infty v}$.

Table 5.2: Classification of the spherical harmonics and $C_{\infty v}$ orbitals in the C_{2v} group.

Symm.[a]	Spherical harmonics (orbitals in $C_{\infty v}$)					
a_1 (1)	s (σ)	p_z (σ)	d_{z^2} (σ)	$d_{x^2-y^2}$ (δ)	f_{z^3} (σ)	$f_{z(x^2-y^2)}$ (δ)
b_1 (2)	p_x (π)	d_{xz} (π)	$f_{x(z^2-y^2)}$ (π)	f_{x^3} (ϕ)		
b_2 (3)	p_y (π)	d_{yz} (π)	$f_{y(z^2-x^2)}$ (π)	f_{y^3} (ϕ)		
a_2 (4)	d_{xy} (δ)	f_{xyz} (δ)				

[a] In parenthesis the number of the symmetry in MOLCAS. It depends on the generators used in `SEWARD`.

In symmetry a_1 we find both σ and δ orbitals. When the calculation is performed in C_{2v} symmetry all the orbitals of a_1 symmetry can mix because they belong to the same representation, but this is not correct for $C_{\infty v}$. The total symmetry must be kept $C_{\infty v}$ and therefore the δ orbitals should not be allowed to rotate and mix with the σ orbitals. The same is true in the b_1 and b_2 symmetries with the π and ϕ orbitals, while in a_2 symmetry this problem does not exist because it has only δ orbitals (with a basis set up to f functions).

The tool to restrict possible orbital rotations is the option SUPSYM in the RASSCF program. It is important to start with clean orbitals belonging to the actual symmetry, that is, without unwanted mixing.

But the problems with the symmetry are not solved with the SUPSYM option only. Orbitals belonging to different components of a degenerate representation should also be equivalent. For example: the π orbitals in b_1 and b_2 symmetries should have the same shape, and the same is true for the δ orbitals in a_1 and a_2 symmetries. This can only be partly achieved in the RASSCF code. The input option AVERAGE will average the density matrices for representations b_1 and b_2 (π and ϕ orbitals), thus producing equivalent orbitals. The present version does not, however, average the δ orbital densities in representations a_1 and a_2 (note that this problem does not occur for electronic states with an equal occupation of the two components of a degenerate set, for example Σ states). A safe way to obtain totally symmetric orbitals is to reduce the symmetry to C_1 (or C_s in the homonuclear case) and perform a state-average calculation for the degenerate components.

We need an equivalence table to know the correspondence of the symbols for the functions in MOLCAS to the spherical harmonics (SH):

Table 5.3: MOLCAS labeling of the spherical harmonics.

MOLCAS	SH	MOLCAS	SH	MOLCAS	SH
1s	s	3d2+	$d_{x^2-y^2}$	4f3+	f_{x^3}
2px	p_x	3d1+	d_{xz}	4f2+	$f_{z(x^2-y^2)}$
2pz	p_z	3d0	d_{z^2}	4f1+	$f_{x(z^2-y^2)}$
2py	p_y	3d1-	d_{yz}	4f0	f_{z^3}
		3d2-	d_{xy}	4f1-	$f_{y(z^2-x^2)}$
				4f2-	f_{xyz}
				4f3-	f_{y^3}

We begin by performing a SCF calculation and analyzing the resulting orbitals. The employed bond distance is close to the experimental equilibrium bond length for the ground state [22]. Observe in the following SEWARD input that the symmetry generators, planes yz and xz, lead to a C_{2v} representation. In the SCF input we have used the option OCCNUMBERS which allows specification of occupation numbers other than 0 or 2. It is still the closed shell SCF energy functional which is optimized, so the obtained SCF energy has no physical meaning. However, the computed orbitals are somewhat better for open shell cases as NiH. The energy of the virtual orbitals is set to zero due to the use of the IVO option. The order of the orbitals may change in different computers and versions of the code.

```
␣&SEWARD␣&END
Title
␣NiH␣G.S.
Symmetry
X␣Y
Basis␣set
Ni.ANO-L...5s4p3d1f.
```

```
Ni␣␣␣␣0.00000␣␣␣0.00000␣␣␣0.000000␣␣␣Bohr
End␣of␣basis
Basis␣set
H.ANO-L...3s2p.
H␣␣␣␣␣0.000000␣␣0.000000␣␣2.747000␣␣␣Bohr
End␣of␣basis
End␣of␣Input
␣&SCF␣&END
TITLE
␣NiH␣G.S.
OCCUPIED
␣8␣3␣3␣1
OCCNumber
2.0␣2.0␣2.0␣2.0␣2.0␣2.0␣2.0␣2.0
2.0␣2.0␣2.0
2.0␣2.0␣2.0
1.0
END␣OF␣INPUT
```

```
   SCF orbitals + arbitrary occupations

   Molecular orbitals for symmetry species 1

   ORBITAL        4         5         6         7         8         9        10
   ENERGY      -4.7208   -3.1159    -.5513    -.4963    -.3305     .0000     .0000
   OCC. NO.     2.0000    2.0000    2.0000    2.0000    2.0000     .0000     .0000

 1 NI   1s0      .0000     .0001     .0000    -.0009     .0019     .0112     .0000
 2 NI   1s0      .0002     .0006     .0000    -.0062     .0142     .0787     .0000
 3 NI   1s0     1.0005    -.0062     .0000    -.0326     .0758     .3565     .0000
 4 NI   1s0      .0053     .0098     .0000     .0531    -.4826     .7796     .0000
 5 NI   1s0     -.0043    -.0032     .0000     .0063    -.0102    -.0774     .0000
 6 NI   2pz      .0001     .0003     .0000    -.0015     .0029     .0113     .0000
 7 NI   2pz     -.0091    -.9974     .0000    -.0304     .0622     .1772     .0000
 8 NI   2pz      .0006     .0013     .0000     .0658    -.1219     .6544     .0000
 9 NI   2pz      .0016     .0060     .0000     .0077    -.0127    -.0646     .0000
10 NI   3d0     -.0034     .0089     .0000     .8730     .4270     .0838     .0000
11 NI   3d0      .0020     .0015     .0000     .0068     .0029     .8763     .0000
12 NI   3d0      .0002     .0003     .0000    -.0118    -.0029    -.7112     .0000
13 NI   3d2+     .0000     .0000    -.9986     .0000     .0000     .0000     .0175
14 NI   3d2+     .0000     .0000     .0482     .0000     .0000     .0000     .6872
15 NI   3d2+     .0000     .0000     .0215     .0000     .0000     .0000    -.7262
16 NI   4f0      .0002     .0050     .0000    -.0009    -.0061     .0988     .0000
17 NI   4f2+     .0000     .0000     .0047     .0000     .0000     .0000    -.0033
18 H    1s0     -.0012    -.0166     .0000     .3084    -.5437    -.9659     .0000
19 H    1s0     -.0008    -.0010     .0000    -.0284    -.0452    -.4191     .0000
20 H    1s0      .0014     .0007     .0000     .0057     .0208     .1416     .0000
21 H    2pz      .0001     .0050     .0000    -.0140     .0007     .5432     .0000
22 H    2pz      .0008    -.0006     .0000     .0060    -.0093     .2232     .0000

   ORBITAL       11        12        13        14        15        16        18
   ENERGY        .0000     .0000     .0000     .0000     .0000     .0000     .0000
   OCC. NO.      .0000     .0000     .0000     .0000     .0000     .0000     .0000

 1 NI   1s0     -.0117    -.0118     .0000     .0025     .0218    -.0294     .0000
 2 NI   1s0     -.0826    -.0839     .0000     .0178     .1557    -.2087     .0000
 3 NI   1s0     -.3696    -.3949     .0000     .0852     .7386    -.9544     .0000
 4 NI   1s0    -1.3543   -1.1537     .0000     .3672    2.3913   -2.8883     .0000
 5 NI   1s0     -.3125     .0849     .0000   -1.0844     .3670    -.0378     .0000
 6 NI   2pz     -.0097    -.0149     .0000     .0064     .0261    -.0296     .0000
 7 NI   2pz     -.1561    -.2525     .0000     .1176     .4515    -.4807     .0000
 8 NI   2pz     -.3655   -1.0681     .0000     .0096    1.7262   -2.9773     .0000
 9 NI   2pz    -1.1434    -.0140     .0000    -.1206     .2437    -.9573     .0000
10 NI   3d0     -.1209    -.2591     .0000     .2015     .5359    -.4113     .0000
11 NI   3d0     -.3992    -.3952     .0000     .1001     .3984    -.9939     .0000
```

```
12 NI  3d0     -.1546    -.1587     .0000    -.1676    -.2422    -.4852     .0000
13 NI  3d2+     .0000     .0000    -.0048     .0000     .0000     .0000    -.0498
14 NI  3d2+     .0000     .0000    -.0017     .0000     .0000     .0000    -.7248
15 NI  3d2+     .0000     .0000     .0028     .0000     .0000     .0000    -.6871
16 NI  4f0     -.1778   -1.0717     .0000    -.0233     .0928    -.0488     .0000
17 NI  4f2+     .0000     .0000   -1.0000     .0000     .0000     .0000    -.0005
18 H   1s0     1.2967    1.5873     .0000    -.3780   -2.7359    3.8753     .0000
19 H   1s0     1.0032     .4861     .0000     .3969    -.9097    1.8227     .0000
20 H   1s0     -.2224    -.2621     .0000     .1872     .0884    -.7173     .0000
21 H   2pz     -.1164    -.4850     .0000     .3388    1.1689    -.4519     .0000
22 H   2pz     -.1668    -.0359     .0000     .0047     .0925    -.3628     .0000

   Molecular orbitals for symmetry species 2

   ORBITAL        2         3         4         5         6         7
   ENERGY     -3.1244    -.5032     .0000     .0000     .0000     .0000
   OCC. NO.    2.0000    2.0000     .0000     .0000     .0000     .0000

 1 NI  2px     -.0001     .0001     .0015     .0018     .0012    -.0004
 2 NI  2px     -.9999     .0056     .0213     .0349     .0235    -.0054
 3 NI  2px     -.0062    -.0140     .1244    -.3887     .2021    -.0182
 4 NI  2px      .0042     .0037     .0893     .8855    -.0520     .0356
 5 NI  3d1+     .0053     .9993     .0268     .0329     .0586     .0005
 6 NI  3d1+    -.0002    -.0211    -.5975     .1616     .1313     .0044
 7 NI  3d1+    -.0012    -.0159     .7930     .0733     .0616     .0023
 8 NI  4f1+     .0013    -.0049     .0117     .1257    1.0211    -.0085
 9 NI  4f3+    -.0064     .0000    -.0003    -.0394     .0132     .9991
10 H   2px     -.0008     .0024    -.0974    -.1614    -.2576    -.0029
11 H   2px      .0003    -.0057    -.2060    -.2268    -.0768    -.0079

   Molecular orbitals for symmetry species 3

   ORBITAL        2         3         4         5         6         7
   ENERGY     -3.1244    -.5032     .0000     .0000     .0000     .0000
   OCC. NO.    2.0000    2.0000     .0000     .0000     .0000     .0000

 1 NI  2py     -.0001     .0001    -.0015     .0018     .0012     .0004
 2 NI  2py     -.9999     .0056    -.0213     .0349     .0235     .0054
 3 NI  2py     -.0062    -.0140    -.1244    -.3887     .2021     .0182
 4 NI  2py      .0042     .0037    -.0893     .8855    -.0520    -.0356
 5 NI  3d1-     .0053     .9993    -.0268     .0329     .0586    -.0005
 6 NI  3d1-    -.0002    -.0211     .5975     .1616     .1313    -.0044
 7 NI  3d1-    -.0012    -.0159    -.7930     .0733     .0616    -.0023
 8 NI  4f3-     .0064     .0000    -.0003     .0394    -.0132     .9991
 9 NI  4f1-     .0013    -.0049    -.0117     .1257    1.0211     .0085
10 H   2py     -.0008     .0024     .0974    -.1614    -.2576     .0029
11 H   2py      .0003    -.0057     .2060    -.2268    -.0768     .0079

   Molecular orbitals for symmetry species 4

   ORBITAL        1         2         3         4
   ENERGY      -.0799     .0000     .0000     .0000
   OCC. NO.    1.0000     .0000     .0000     .0000

 1 NI  3d2-    -.9877    -.0969     .0050    -.1226
 2 NI  3d2-    -.1527     .7651     .0019     .6255
 3 NI  3d2-    -.0332    -.6365    -.0043     .7705
 4 NI  4f2-     .0051    -.0037    1.0000     .0028
```

In difficult situations it can be useful to employ the AUFBAU option of the `SCF` program. Including this option, the subsequent classification of the orbitals in the different symmetry representations can be avoided. The program will look for the lowest-energy solution and will provide with a final occupation. This option must be used with caution. It is only expected to work in clear closed-shell situations.

We have only printed the orbitals most relevant to the following discussion. Starting with symmetry 1 (a_1) we observe that the orbitals are not mixed at all. Using a basis set contracted to Ni $5s4p3d1f$ / H $3s2p$ in symmetry a_1 we obtain 18 σ molecular orbitals (combinations from eight atomic s functions, six p_z functions, three d_{z^2} functions, and one f_{z^3} function) and four δ orbitals (from three $d_{x^2-y^2}$ functions and one $f_{z(x^2-y^2)}$ function). Orbitals 6, 10, 13, and 18 are formed by contributions from the three $d_{x^2-y^2}$ and one $f_{z(x^2-y^2)}$ δ functions, while the contributions of the remaining harmonics are zero. These orbitals are δ orbitals and should not mix with the remaining a_1 orbitals. The same situation occurs in symmetries b_1 and b_2 (2 and 3) but in this case we observe an important mixing among the orbitals. Orbitals $7b_1$ and $7b_2$ have main contributions from the harmonics 4f3+ (f_{x^3}) and 4f3- (f_{y^3}), respectively. They should be pure ϕ orbitals and not mix at all with the remaining π orbitals.

The first step is to evaluate the importance of the mixings for future calculations. Strictly, any kind of mixing should be avoided. If g functions are used, for instance, new contaminations show up. But, undoubtedly, not all mixings are going to be equally important. If the rotations occur among occupied or active orbitals the influence on the results is going to be larger than if they are high secondary orbitals. NiH is one of these cases. The ground state of the molecule is $^2\Delta$. It has two components and we can therefore compute it by placing the single electron in the d_{xy} orbital (leading to a state of a_2 symmetry in C_{2v}) or in the $d_{x^2-y^2}$ orbital of the a_1 symmetry. Both are δ orbitals and the resulting states will have the same energy provided that no mixing happens. In the a_2 symmetry no mixing is possible because it is only composed of δ orbitals but in a_1 symmetry the σ and δ orbitals can rotate. It is clear that this type of mixing will be more important for the calculation than the mixing of π and ϕ orbitals. However it might be necessary to prevent it. Because in the SCF calculation no high symmetry restriction was imposed on the orbitals, orbitals 2 and 4 of the b_1 and b_2 symmetries have erroneous contributions of the 4f3+ and 4f3- harmonics, and they are occupied or active orbitals in the following CASSCF calculation.

To use the supersymmetry (SUPSYM) option we must start with proper orbitals. In this case the a_1 orbitals are symmetry adapted (within the printed accuracy) but not the b_1 and b_2 orbitals. Orbitals $7b_1$ and $7b_2$ must have zero coefficients for all the harmonics except for 4f3+ and 4f3-, respectively. The remaining orbitals of these symmetries (even those not shown) must have zero in the coefficients corresponding to 4f3+ or 4f3-. To clean the orbitals the option CLEANUP of the `RASSCF` program can be used.

Once the orbitals are properly symmetrized we can perform CASSCF calculations on different electronic states. Deriving the types of the molecular electronic states resulting from the electron configurations is not simple in many cases. In general, for a given electronic configuration several electronic states of the molecule will result. Wigner and Witmer derived rules for determining what types of molecular states result from given states of the separated atoms. In chapter VI of reference [24] it is possible to find the tables of the resulting electronic states once the different couplings and the Pauli principle have been applied.

In the present CASSCF calculation we have chosen the active space ($3d$, $4d$, σ, σ^*) with all the 11 valence electrons active. If we consider $4d$ and σ^* as weakly occupied correlating orbitals, we are left with $3d$ and σ (six orbitals), which are to be occupied with 11 electrons. Since the bonding orbital σ (composed mainly of Ni $4s$ and H $1s$) will be doubly occupied in all low lying electronic states, we are left with nine electrons to occupy the $3d$ orbitals. There is thus one hole, and the possible electronic states are: $^2\Sigma^+$, $^2\Pi$, and $^2\Delta$, depending on the orbital where the hole is located. Taking Table 5.4 into account we observe that we have two low-lying electronic states in symmetry 1 (A_1): $^2\Sigma^+$ and $^2\Delta$, and one in each of the other three symmetries: $^2\Pi$ in symmetries 2 (B_1) and 3 (B_2), and $^2\Delta$ in symmetry 4 (A_2). It is not immediately obvious which of these states is the ground state as they are close in energy. It may therefore be necessary to study all of them. It has been found at different levels of theory that the NiH has a $^2\Delta$ ground state [22].

We continue by computing the $^2\Delta$ ground state. The previous SCF orbitals will be the initial orbitals for the CASSCF calculation. First we need to know in which C_{2v} symmetry or symmetries we can compute a Δ state. In the symmetry tables it is determined how the species of the linear molecules are resolved into those of lower symmetry (depends also on the orientation of the molecule). In Table 5.4 is listed the assignment of the different symmetries for the molecule placed on the z axis.

The Δ state has two degenerate components in symmetries a_1 and a_2. Two CASSCF calculations can be performed, one computing the first root of a_2 symmetry and the second for the first root of a_1 symmetry. The `RASSCF` input for the state of a_2 symmetry would be:

```
␣&RASSCF␣&END
Title
␣NiH␣2Delta␣CAS␣s,␣s*,␣3d,␣3d’.
Symmetry
␣␣␣␣4
Spin
␣␣␣␣2
Nactel
␣␣␣11␣␣␣␣0␣␣␣␣0
Inactive
␣␣␣␣5␣␣␣␣2␣␣␣␣2␣␣␣␣0
Ras2
␣␣␣␣6␣␣␣␣2␣␣␣␣2␣␣␣␣2
Thrs
1.0E-07,1.0E-05,1.0E-05
Cleanup
1
␣␣4␣6␣10␣13␣18
␣18␣1␣2␣3␣4␣5␣6␣7␣8␣9␣10␣11␣12␣16␣18␣19␣20␣21␣22
␣␣4␣13␣14␣15␣17
1
␣␣1␣7
␣␣10␣1␣2␣3␣4␣5␣6␣7␣8␣10␣11
␣␣1␣9
1
␣␣1␣7
␣␣10␣1␣2␣3␣4␣5␣6␣7␣9␣10␣11
␣␣1␣8
0
Supsym
1
␣␣␣4␣6␣10␣13␣18
1
␣␣␣1␣7
1
␣␣␣1␣7
0
*Average
*1␣2␣3
Iter
50,25
LumOrb
End␣of␣Input
```

The corresponding input for symmetry a_1 will be identical except for the SYMMETRY keyword

```
Symmetry
␣␣␣␣1
```

Table 5.4: Resolution of the $C_{\infty v}$ species in the C_{2v} species.

State symmetry $C_{\infty v}$	State symmetry C_{2v}
Σ^+	A_1
Σ^-	A_2
Π	$B_1 + B_2$
Δ	$A_1 + A_2$
Φ	$B_1 + B_2$
Γ	$A_1 + A_2$

In the `RASSCF` inputs the CLEAnup option will take the initial orbitals (SCF here) and will place zeroes in all the coefficients of orbitals 6, 10, 13, and 18 in symmetry 1, except in coefficients 13, 14, 15, and 17. Likewise all coefficients 13, 14, 15, and 17 of the remaining a_1 orbitals will be set to zero. The same procedure is used in symmetries b_1 and b_2. Once cleaned, and because of the SUPSymmetry option, the δ orbitals 6, 10, 13, and 18 of a_1 symmetry will only rotate among themselves and they will not mix with the remaining a_1 σ orbitals. The same holds true for ϕ orbitals $7b_1$ and $7b_2$ in their respective symmetries.

Orbitals can change order during the calculation. *MOLCAS* incorporates a procedure to check the nature of the orbitals in each iteration. Therefore the right behavior of the SUPSym option is guaranteed during the calculation. The procedure can have problems if the initial orbitals are not symmetrized properly. Therefore, the output with the final results should be checked to compare the final order of the orbitals and the final labeling of the SUPSym matrix.

The AVERage option would average the density matrices of symmetries 2 and 3, corresponding to the Π and Φ symmetries in $C_{\infty v}$. In this case it is not necessary to use the option because the two components of the degenerate sets in symmetries b_1 and b_2 have the same occupation and therefore they will have the same shape. The use of the option in a situation like this ($^2\Delta$ and $^2\Sigma^+$ states) leads to convergence problems. The symmetry of the orbitals in symmetries 2 and 3 is retained even if the AVERage option is not used.

The output for the calculation on symmetry 4 (a_2) contains the following lines:

```
      Convergence after  29 iterations
       30   2    2    1 -1507.59605678    -.23E-11   3   9 1  -.68E-06  -.47E-05

                                          Wave function printout:
 occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

      printout of CI-coefficients larger than   .05 for root   1
      energy=  -1507.596057
      conf/sym  111111 22 33 44     Coeff  Weight
         15834  222000 20 20 u0    .97979  .95998
         15838  222000 ud ud u0    .05142  .00264
         15943  2u2d00 ud 20 u0   -.06511  .00424
         15945  2u2d00 20 ud u0    .06511  .00424
         16212  202200 20 20 u0   -.05279  .00279
         16483  u220d0 ud 20 u0   -.05047  .00255
         16485  u220d0 20 ud u0    .05047  .00255
```

```
Natural orbitals and occupation numbers for root  1
sym 1:   1.984969   1.977613   1.995456    .022289    .014882    .005049
sym 2:   1.983081    .016510
sym 3:   1.983081    .016510
sym 4:    .993674    .006884
```

The state is mainly (weight 96%) described by a single configuration (configuration number 15834) which placed one electron on the first active orbital of symmetry 4 (a_2) and the remaining electrons are paired. A close look to this orbital indicates that is has a coefficient -.9989 in the first 3d2- ($3d_{xy}$) function and small coefficients in the other functions. This results clearly indicate that we have computed the $^2\Delta$ state as the lowest root of that symmetry. The remaining configurations have negligible contributions. If the orbitals are properly symmetrized, all configurations will be compatible with a $^2\Delta$ electronic state.

The calculation of the first root of symmetry 1 (a_1) results:

```
Convergence after  15 iterations
 16   2    3    1 -1507.59605678    -.19E-10   8  15 1   .35E-06  -.74E-05

                                  Wave function printout:
 occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

       printout of CI-coefficients larger than   .05 for root   1
       energy=  -1507.596057
       conf/sym  111111 22 33 44     Coeff  Weight
          40800  u22000 20 20 20   -.97979  .95998
          42400  u02200 20 20 20    .05280  .00279

       Natural orbitals and occupation numbers for root  1
       sym 1:    .993674   1.977613   1.995456    .022289    .006884    .005049
       sym 2:   1.983081    .016510
       sym 3:   1.983081    .016510
       sym 4:   1.984969    .014882
```

We obtain the same energy as in the previous calculation. Here the dominant configuration places one electron on the first active orbital of symmetry 1 (a_1). It is important to remember that the orbitals are not ordered by energies or occupations into the active space. This orbital has also the coefficient -.9989 in the first 3d2$-$ ($3d_{x^2-y^2}$) function. We have then computed the other component of the $^2\Delta$ state. As the δ orbitals in different C_{2v} symmetries are not averaged by the program it could happen (not in the present case) that the two energies differ slightly from each other.

The consequences of not using the SUPSYM option are not extremely severe in the present example. If you perform a calculation without the option, the obtained energy is:

```
Convergence after  29 iterations
 30   2    2    1 -1507.59683719    -.20E-11   3   9 1  -.69E-06  -.48E-05
```

As it is a broken symmetry solution the energy is lower than in the other case. This is a typical behavior. If we were using an exact wave function it would have the right symmetry properties, but approximated wave functions do not necessarily fulfil this condition. So, more flexibility leads to lower energy solutions which have broken the orbital symmetry.

If in addition to the $^2\Delta$ state we want to compute the lowest $^2\Sigma^+$ state we can use the adapted orbitals from any of the $^2\Delta$ state calculations and use the previous `RASSCF` input without the CLEANUP option. The orbitals have not changed place in this example. If they do, one has to change the labels in the SUPSYM option. The simplest way to compute the lowest excited

$^2\Sigma^+$ state is having the unpaired electron in one of the σ orbitals because none of the other configurations, δ^3 or π^3, leads to the $^2\Sigma^+$ term. However, there are more possibilities such as the configuration $\sigma^1\sigma^1\sigma^1$; three nonequivalent electrons in three σ orbitals. In actuality the lowest $^2\Sigma^+$ state must be computed as a doublet state in symmetry A$_1$. Therefore, we set the symmetry in the RASSCF to 1 and compute the second root of the symmetry (the first was the $^2\Delta$ state):

```
CIRoot
1␣2
2
```

Of course the SUPSym option must be maintained. The use of CIROot indicates that we are computing the second root of that symmetry. The obtained result:

```
      Convergence after  33 iterations
        9   2    3    2 -1507.58420263    -.44E-10   2  11 2  -.12E-05   .88E-05

                                      Wave function printout:
 occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

      printout of CI-coefficients larger than   .05 for root   1
      energy=  -1507.584813
      conf/sym  111111 22 33 44     Coeff  Weight
         40800  u22000 20 20 20   -.97917  .95877

      printout of CI-coefficients larger than   .05 for root   2
      energy=  -1507.584203
      conf/sym  111111 22 33 44     Coeff  Weight
         40700  2u2000 20 20 20    .98066  .96169

      Natural orbitals and occupation numbers for root  2
      sym 1:   1.983492    .992557   1.995106    .008720    .016204    .004920
      sym 2:   1.983461    .016192
      sym 3:   1.983451    .016192
      sym 4:   1.983492    .016204
```

As we have used two as the dimension of the CI matrix employed in the CI Davidson procedure we obtain the wave function of two roots, although the optimized root is the second. Root 1 places one electron in the first active orbital of symmetry one, which is a 3d2+ ($3d_{x^2-y^2}$) δ orbital. Root 2 places the electron in the second active orbital, which is a σ orbital with a large coefficient (.9639) in the first 3d0 ($3d_{z^2}$) function of the nickel atom. We have therefore computed the lowest $^2\Sigma^+$ state. The two $^2\Sigma^+$ states resulting from the configuration with the three unpaired σ electrons is higher in energy at the CASSCF level. If the second root of symmetry a$_1$ had not been a $^2\Sigma^+$ state we would have to study higher roots of the same symmetry.

It is important to remember that the active orbitals are not ordered at all within the active space. Therefore, their order might vary from calculation to calculation and, in addition, no conclusions about the orbital energy, occupation or any other information can be obtained from the order of the active orbitals.

We can compute also the lowest $^2\Pi$ excited state. The simplest possibility is having the configuration π^3, which only leads to one $^2\Pi$ state. The unpaired electron will be placed in either one b$_1$ or one b$_2$ orbital. That means that the state has two degenerate components and we can compute it equally in both symmetries. There are more possibilities, such as the configuration $\pi^3\sigma^1\sigma^1$ or the configuration $\pi^3\sigma^1\delta^1$. The resulting $^2\Pi$ state will always have two degenerate components in symmetries b_1 and b_2, and therefore it is the wave function analysis which gives us the information of which configuration leads to the lowest $^2\Pi$ state.

For NiH it turns out to be non trivial to compute the $^2\Pi$ state. Taking as initial orbitals the previous SCF orbitals and using any type of restriction such as the CLEAnup, SUPSym or AVERage options lead to severe convergence problems like these:

```
 45   9   17    1 -1507.42427683    -.65E-02   6  18 1  -.23E-01  -.15E+00
 46   5   19    1 -1507.41780710     .65E-02   8  15 1   .61E-01  -.15E+00
 47   9   17    1 -1507.42427683    -.65E-02   6  18 1  -.23E-01  -.15E+00
 48   5   19    1 -1507.41780710     .65E-02   8  15 1   .61E-01  -.15E+00
 49   9   17    1 -1507.42427683    -.65E-02   6  18 1  -.23E-01  -.15E+00
 50   5   19    1 -1507.41780710     .65E-02   8  15 1   .61E-01  -.15E+00

No convergence after  50 iterations
 51   9   19    1 -1507.42427683    -.65E-02   6  18 1  -.23E-01  -.15E+00
```

The calculation, however, converges in an straightforward way if none of those tools are used:

```
      Convergence after  33 iterations
       34   2    2    1 -1507.58698677    -.23E-12   3   8 2  -.72E-06  -.65E-05

                                   Wave function printout:
 occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

      printout of CI-coefficients larger than   .05 for root   1
      energy=  -1507.586987
      conf/sym  111111 22 33 44     Coeff  Weight
         15845  222000 u0 20 20    .98026  .96091
         15957  2u2d00 u0 ud 20    .05712  .00326
         16513  u220d0 u0 20 ud   -.05131  .00263

      Natural orbitals and occupation numbers for root  1
      sym 1:   1.984111   1.980077   1.995482    .019865    .015666    .004660
      sym 2:    .993507    .007380
      sym 3:   1.982975    .016623
      sym 4:   1.983761    .015892
```

The π (and ϕ) orbitals, both in symmetries b_1 and b_2, are, however, differently occupied and therefore are not equal as they should be:

```
   Molecular orbitals for sym species 2      Molecular orbitals for symmetry species 3
   ORBITAL         3         4               ORBITAL         3         4
   ENERGY        .0000     .0000             ENERGY        .0000     .0000
   OCC. NO.      .9935     .0074             OCC. NO.     1.9830     .0166

 1 NI   2px      .0001     .0002           1 NI   2py      .0018    -.0001
 2 NI   2px      .0073     .0013           2 NI   2py      .0178    -.0002
 3 NI   2px     -.0155     .0229           3 NI   2py     -.0197    -.0329
 4 NI   2px      .0041     .0227           4 NI   2py      .0029    -.0254
 5 NI   3d1+     .9990    -.0199           5 NI   3d1-     .9998    -.0131
 6 NI   3d1+    -.0310    -.8964           6 NI   3d1-     .0128     .9235
 7 NI   3d1+    -.0105     .4304           7 NI   3d1-     .0009    -.3739
 8 NI   4f1+    -.0050     .0266           8 NI   4f3-     .0001    -.0003
 9 NI   4f3+     .0001     .0000           9 NI   4f1-    -.0050    -.0177
10 H    2px      .0029    -.0149          10 H    2py      .0009     .0096
11 H    2px     -.0056    -.0003          11 H    2py     -.0094    -.0052
```

Therefore what we have is a symmetry broken solution. To obtain a solution which is not of broken nature the π and ϕ orbitals must be equivalent. The tool to obtain equivalent orbitals is the AVERage option, which averages the density matrices of symmetries b_1 and

b_2. But starting with any of the preceding orbitals and using the AVERage option lead again to convergence problems. It is necessary to use better initial orbitals; orbitals which have already equal orbitals in symmetries b_1 and b_2. One possibility is to perform a SCF calculation on the NiH cation explicitly indicating occupation one in the two higher occupied π orbitals (symmetries 2 and 3):

```
␣&SCF␣&END
TITLE
␣NiH␣cation
OCCUPIED
␣8␣3␣3␣1
OCCNO
2.0␣2.0␣2.0␣2.0␣2.0␣2.0␣2.0␣2.0
2.0␣2.0␣1.0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣<--␣Note␣the␣extra␣occupation
2.0␣2.0␣1.0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣<--␣Note␣the␣extra␣occupation
2.0
IVO
END␣OF␣INPUT
```

It can take some successive steps to obtain a converged calculation using the CLEAnup, SUPSym, and AVERage options. The calculation with a single root did not converge clearly. We obtained, however, a converged result for the lowest $^2\Pi$ state of NiH by computing two averaged CASSCF roots and setting a weight of 90% for the first root using the keyword:

```
CIROot
␣2␣2
␣1␣2
␣9␣1
```

```
                            Wave function printout:
occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

     printout of CI-coefficients larger than   .05 for root   1
     energy=  -1507.566492
     conf/sym  111111 22 33 44     Coeff  Weight
         4913  222u00 20 d0 u0   -.05802  .00337
        15845  222000 u0 20 20    .97316  .94703
        15953  2u2d00 u0 20 20    .05763  .00332
        16459  2u20d0 u0 20 ud   -.05283  .00279

     Natural orbitals and occupation numbers for root  1
     sym 1:   1.972108   1.982895   1.998480    .028246    .016277    .007159
     sym 2:    .997773    .007847
     sym 3:   1.978019    .016453
     sym 4:   1.978377    .016366
```

The energy of the different states (only the first one shown above) is printed on the top of their configuration list. The converged energy is simply an average energy. The occupation numbers obtained in the section of the `RASSCF` output printed above are the occupation numbers of the natural orbitals of the corresponding root. They differ from the occupation numbers printed in the molecular orbital section where we have pseudonatural molecular orbitals and average occupation numbers. On top of each of the valence π orbitals an average occupation close to 1.5e will be printed; this is a consequence of the the averaging procedure.

The results obtained are only at the CASSCF level. Additional effects have to be considered and included. The most important of them is the dynamical correlation effect which can be added by computing, for instance, the CASPT2 energies. The reader can find a detailed explanation of the different approaches in ref. [22], and a careful discussion of their consequences and solutions in ref. [25].

We are going, however, to point out some details. In the first place the basis set must include up to g functions for the transition metal atom and up to d functions for the hydrogen. Relativistic effects must be taken into account, at least in a simple way as a first order correction. The keyword RELINT must be then included in the `SEWARD` input to compute the mass-velocity and one-electron Darwin contact term integrals and obtain a first-order correction to the energy with respect to relativistic effects at the CASSCF level in the `RASSCF` output. Scalar relativistic effects can be also included according the Douglas-Kroll or the Barysz-Sadlej-Snijders transformations, as it will be explained in section 5.7.

The CASPT2 input needed to compute the second-order correction to the energy will include the number of the CASSCF root to compute. For instance, for the first root of each symmetry:

```
␣&CASPT2␣&END
Title
␣NiH
Frozen
5␣2␣2␣0
Maxit
30
Lroot
1
End␣of␣input
```

The number of frozen orbitals taken by `CASPT2` will be that specified in the `RASSCF` input except if this is changed in the `CASPT2` input. In the perturbative step we have frozen all the occupied orbitals except the active ones. This is motivated by the desire to include exclusively the dynamical correlation related to the valence electrons. In this way we neglect correlation between core electrons, named core-core correlation, and between core and valence electrons, named core-valence correlation. This is not because the calculation is smaller but because of the inclusion of those type of correlation in a calculation designed to treat valence correlation is an inadequate approach. Core-core and core-valence correlation requires additional basis functions of the same spatial extent as the occupied orbitals being correlated, but with additional radial and angular nodes. Since the spatial extent of the core molecular orbitals is small, the exponents of these correlating functions must be much larger than those of the valence optimized basis sets. The consequence is that we must avoid the inclusion of the core electrons in the treatment in the first step. Afterwards, the amount of correlation introduced by the core electrons can be estimated in separated calculations for the different states and those effects added to the results with the valence electrons.

Core-valence correlation effects of the $3s$ and $3p$ nickel shells can be studied by increasing the basis set flexibility by uncontracting the basis set in the appropriate region. There are different possibilities. Here we show the increase of the basis set by four s, four p, and four d functions. f functions contribute less to the description of the $3s$ and $3p$ shells and can be excluded. The uncontracted exponents should correspond to the region where the $3s$ and $3p$ shells present their density maximum. Therefore, first we compute the absolute maxima of the radial distribution of the involved orbitals, then we determine the primitive gaussian functions which have their maxima in the same region as the orbitals and therefore which exponents should be uncontracted. The final basis set will be the valence basis set used before plus the new added functions. In the present example the SEWARD input can be:

```
␣&SEWARD␣&END
Title
␣NiH␣G.S.
Symmetry
X␣Y
*RelInt
Basis␣set
Ni.ANO-L...5s4p3d1f.
Ni␣␣␣␣0.00000␣␣␣0.00000␣␣␣0.000000␣␣␣Bohr
End␣of␣basis
Basis␣set
Ni....4s4p4d.␣/␣Inline
␣0.␣␣2
*␣Additional␣s␣functions
␣4␣4
3.918870␣1.839853␣0.804663␣0.169846
␣1.␣0.␣0.␣0.
␣0.␣1.␣0.␣0.
␣0.␣0.␣1.␣0.
␣0.␣0.␣0.␣1.
*␣Additional␣p␣functions
␣4␣4
2.533837␣1.135309␣0.467891␣0.187156
␣1.␣0.␣0.␣0.
␣0.␣1.␣0.␣0.
␣0.␣0.␣1.␣0.
␣0.␣0.␣0.␣1.
*␣Additional␣d␣functions
␣4␣4
2.551303␣1.128060␣0.475373␣0.182128
␣1.␣0.␣0.␣0.
␣0.␣1.␣0.␣0.
␣0.␣0.␣1.␣0.
␣0.␣0.␣0.␣1.
Nix␣␣␣0.00000␣␣␣0.00000␣␣␣0.000000␣␣␣Bohr
End␣of␣basis
Basis␣set
H.ANO-L...3s2p.
H␣␣␣␣␣0.000000␣␣0.000000␣␣2.747000␣␣␣Bohr
End␣of␣basis
End␣of␣Input
```

We have used a special format to include the additional functions. We include the additional $4s4p4d$ functions for the nickel atom. The additional basis set input must use a dummy label (Nix here), the same coordinates of the original atom, and specify a CHARGE equal to zero, whether in an Inline basis set input as here or by specifically using keyword CHARGE. It is not necessary to include the basis set with the Inline format. A library can be created for this purpose. In this case the label for the additional functions could be:

```
Ni.Uncontracted...4s4p4d.␣/␣AUXLIB
Charge
0
```

and a proper link to AUXLIB should be included in the script (or in the input if one uses AUTO).

Now the CASPT2 is going to be different to include also the correlation related to the $3s, 3p$ shell of the nickel atom. Therefore, we only freeze the $1s, 2s, 2p$ shells:

```
␣&CASPT2␣&END
Title
␣NiH.␣Core-valence.
```

```
Frozen
3 1 1 0
Maxit
30
Lroot
1
End of input
```

A final effect one should study is the basis set superposition error (BSSE). In many cases it is a minor effect but it is an everpresent phenomenon which should be investigated when high accuracy is required, especially in determining bond energies, and not only in cases with weakly interacting systems, as is frequently believed. The most common approach to estimate this effect is the counterpoise correction: the separated fragment energies are computed in the total basis set of the system. For a discussion of this issue see Refs. [25, 26]. In the present example we would compute the energy of the isolated nickel atom using a SEWARD input including the full nickel basis set plus the hydrogen basis set in the hydrogen position but with the charge set to zero. And then the opposite should be done to compute the energy of isolated hydrogen. The BSSE depends on the separation of the fragments and must be estimated at any computed geometry. For instance, the SEWARD input necessary to compute the isolated hydrogen atom at a given distance from the ghost nickel basis set including core uncontracted functions is:

```
!ln -fs $HomeDir/NiH.NewLib AUXLIB
 &SEWARD &END
Title
 NiH. 3s3p + H (BSSE)
Symmetry
X Y
RelInt
Basis set
Ni.ANO-L...5s4p3d1f.
Ni    0.00000   0.00000   0.000000   Bohr
Charge
0.0
End of basis
Basis set
Ni.Uncontracted...4s4p4d. / AUXLIB
Nix   0.00000   0.00000   0.000000   Bohr
Charge
0.0
End of basis
Basis set
H.ANO-L...3s2p.
H     0.000000  0.000000  2.747000   Bohr
End of basis
End of Input
```

Once the energy of each of the fragments with the corresponding ghost basis set of the other fragment is determined, the energies of the completely isolated fragments can be computed and sustracted from those which have the ghost basis sets. Other approaches used to estimate the BSSE effect are discussed in Ref. [25].

The results obtained at the CASPT2 level are close to those obtained by MRCI+Q and ACPF treatments but more accurate. They match well with experiment. The difference is that all the configuration functions (CSFs) of the active space can be included in CASPT2 in the zeroth-order references for the second-order perturbation calculation [22], while the other methods have to restrict the number of configurations.

Calculations of linear molecules become more and more complicated when the number of unpaired electrons increases. In the following sections we will discuss the more complicated situation occurring in the Ni_2 molecule.

5.1.2 A diatomic homonuclear molecule: C_2

C_2 is a classical example of a system where near-degeneracy effects have large amplitudes even near the equilibrium internuclear separation. The biradical character of the ground state of the molecule suggest that a single configurational treatment will not be appropriate for accurate descriptions of the spectroscopic constants [21]. There are two nearly degenerate states: $^1\Sigma_g^+$ and $^3\Pi_u$. The latter was earlier believed to be the ground state, an historical assignment which can be observed in the traditional labeling of the states.

As C_2 is a $D_{\infty h}$ molecule, we have to compute it in D_{2h} symmetry. We make a similar analysis as for the C_{2v} case. We begin by classifying the functions in $D_{\infty h}$ in Table 5.5. The molecule is placed on the z axis.

Table 5.5: Classification of the spherical harmonics in the $D_{\infty h}$ group[a].

Symmetry	Spherical harmonics			
σ_g	s	d_{z^2}		
σ_u	p_z	f_{z^3}		
π_g	d_{xz}	d_{yz}		
π_u	p_x	p_y	$f_{x(z^2-y^2)}$	$f_{y(z^2-x^2)}$
δ_g	$d_{x^2-y^2}$	d_{xy}		
δ_u	f_{xyz}	$f_{z(x^2-y^2)}$		
ϕ_u	f_{x^3}	f_{y^3}		

[a] Functions placed on the symmetry center.

Table 5.6 classifies the functions and orbitals into the symmetry representations of the D_{2h} symmetry. Note that in table 5.6 subindex b stands for bonding combination and a for antibonding combination.

The order of the symmetries, and therefore the number they have in *MOLCAS*, depends on the generators used in the `SEWARD` input. This must be carefully checked at the beginning of any calculation. In addition, the orientation of the molecule on the cartesian axis can change the labels of the symmetries. In Table 5.6 for instance we have used the order and numbering of a calculation performed with the three symmetry planes of the D_{2h} point group (X Y Z in the `SEWARD` input) and the z axis as the intermolecular axis (that is, x and y are equivalent in D_{2h}). Any change in the orientation of the molecule will affect the labels of the orbitals and states. In this case the π orbitals will belong to the b_{3u}, b_{2u}, b_{2g}, and b_{3g} symmetries. For instance, with x as the intermolecular axis b_{3u} and b_{3g} will be replaced by b_{1u} and b_{1g}, respectively, and finally with y as the intermolecular axis b_{1u}, b_{3u}, b_{3g}, and b_{1g} would be the π orbitals.

It is important to remember that *MOLCAS* works with symmetry adapted basis functions. Only the symmetry independent atoms are required in the `SEWARD` input. The remaining ones will be generated by the symmetry operators. This is also the case for the molecular orbitals. *MOLCAS* will only print the coefficients of the symmetry adapted basis functions.

Table 5.6: Classification of the spherical harmonics and $D_{\infty h}$ orbitals in the D_{2h} group[a].

Symm.[b]	Spherical harmonics (orbitals in $D_{\infty h}$)					
a_g (1)	s_b (σ_g)	p_{zb} (σ_g)	d_{z^2b} (σ_g)	$d_{x^2-y^2b}$ (δ_g)	f_{z^3b} (σ_g)	$f_{z(x^2-y^2)b}$ (δ_g)
b_{3u}(2)	p_{xb} (π_u)	d_{xzb} (π_u)	$f_{x(z^2-y^2)b}$ (π_u)	f_{x^3b} (ϕ_u)		
b_{2u}(3)	p_{yb} (π_u)	d_{yzb} (π_u)	$f_{y(z^2-x^2)b}$ (π_u)	f_{y^3b} (ϕ_u)		
b_{1g}(4)	d_{xyb} (δ_g)	f_{xyzb} (δ_g)				
b_{1u}(5)	s_a (σ_u)	p_{za} (σ_u)	d_{z^2a} (σ_u)	$d_{x^2-y^2a}$ (δ_u)	f_{z^3a} (σ_u)	$f_{z(x^2-y^2)a}$ (δ_u)
b_{2g}(6)	p_{ya} (π_g)	d_{yza} (π_g)	$f_{y(z^2-x^2)a}$ (π_g)	f_{y^3a} (ϕ_g)		
b_{3g}(7)	p_{xa} (π_g)	d_{xza} (π_g)	$f_{x(z^2-y^2)a}$ (π_g)	f_{x^3a} (ϕ_g)		
a_u (8)	d_{xya} (δ_u)	f_{xyza} (δ_u)				

[a] Subscripts a and b refer to the bonding and antibonding combination of the AO's, respectively.

[b] In parenthesis the number of the symmetry in MOLCAS. Note that the number and order of the symmetries depend on the generators and the orientation of the molecule.

The necessary information to obtain the complete set of orbitals is contained in the SEWARD output. Consider the case of the a_g symmetry:

```
                     **************************************************
                     ******** Symmetry adapted Basis Functions ********
                     **************************************************

            Irreducible representation : ag
            Basis function(s) of irrep:

  Basis Label        Type   Center Phase Center Phase
    1   C            1s0       1     1      2     1
    2   C            1s0       1     1      2     1
    3   C            1s0       1     1      2     1
    4   C            1s0       1     1      2     1
    5   C            2pz       1     1      2    -1
    6   C            2pz       1     1      2    -1
    7   C            2pz       1     1      2    -1
    8   C            3d0       1     1      2     1
    9   C            3d0       1     1      2     1
   10   C            3d2+      1     1      2     1
   11   C            3d2+      1     1      2     1
   12   C            4f0       1     1      2    -1
   13   C            4f2+      1     1      2    -1
```

The previous output indicates that symmetry adapted basis function 1, belonging to the a_g representation, is formed by the symmetric combination of a s type function centered on atom C and another s type function centered on the redundant center 2, the second carbon atom. Combination $s + s$ constitutes a bonding σ_g-type orbital. For the p_z function however the combination must be antisymmetric. It is the only way to make the p_z orbitals overlap and form a bonding orbital of a_g symmetry. Similar combinations are obtained for the remaining basis sets of the a_g and other symmetries.

The molecular orbitals will be combinations of these symmetry adapted functions. Consider the a_g orbitals:

```
     SCF orbitals

     Molecular orbitals for symmetry species 1

     ORBITAL         1          2          3          4          5          6
```

```
    ENERGY    -11.3932   -1.0151    -.1138     .1546     .2278     .2869
    OCC. NO.    2.0000    2.0000     .0098     .0000     .0000     .0000

  1 C   1s0     1.4139    -.0666    -.0696     .2599     .0626     .0000
  2 C   1s0      .0003    1.1076    -.6517    1.0224     .4459     .0000
  3 C   1s0      .0002    -.0880    -.2817     .9514     .0664     .0000
  4 C   1s0      .0000    -.0135    -.0655     .3448    -.0388     .0000
  5 C   2pz     -.0006    -.2581   -1.2543    1.1836     .8186     .0000
  6 C   2pz      .0000     .1345    -.0257    2.5126    1.8556     .0000
  7 C   2pz      .0005    -.0192    -.0240     .7025     .6639     .0000
  8 C   3d0      .0003     .0220    -.0005    -.9719     .2430     .0000
  9 C   3d0     -.0001    -.0382    -.0323    -.8577     .2345     .0000
 10 C   3d2+     .0000     .0000     .0000     .0000     .0000    -.7849
 11 C   3d2+     .0000     .0000     .0000     .0000     .0000    -.7428
 12 C   4f0     -.0002    -.0103    -.0165     .0743     .0081     .0000
 13 C   4f2+     .0000     .0000     .0000     .0000     .0000    -.0181
```

In MOLCAS outputs only 13 coefficients for orbital are going to be printed because they are the coefficients of the symmetry adapted basis functions. If the orbitals were not composed by symmetry adapted basis functions they would have, in this case, 26 coefficients, two for type of function (following the scheme observed above in the `SEWARD` output), symmetrically combined the *s* and *d* functions and antisymmetrically combined the *p* and *f* functions.

To compute $D_{\infty h}$ electronic states using the D_{2h} symmetry we need to go to the symmetry tables and determine how the species of the linear molecules are resolved into those of lower symmetry (this depends also on the orientation of the molecule [24]). Table 5.7 lists the case of a $D_{\infty h}$ linear molecule with z as the intermolecular axis.

Table 5.7: Resolution of the $D_{\infty h}$ species in the D_{2h} species.

State symmetry $D_{\infty h}$	State symmetry D_{2h}
Σ_g^+	A_g
Σ_u^+	B_{1u}
Σ_g^-	B_{1g}
Σ_u^-	A_u
Π_g	$B_{2g} + B_{3g}$
Π_u	$B_{2u} + B_{3u}$
Δ_g	$A_g + B_{1g}$
Δ_u	$A_u + B_{1u}$
Φ_g	$B_{2g} + B_{3g}$
Φ_u	$B_{2u} + B_{3u}$
Γ_g	$A_g + B_{1g}$
Γ_u	$A_u + B_{1u}$

To compute the ground state of C_2, a ${}^1\Sigma_g^+$ state, we will compute a singlet state of symmetry A_g (1 in this context). The input files for a CASSCF calculation on the C_2 ground state will be:

```
␣&SEWARD␣&END
Title
␣C2
Symmetry
␣␣X␣␣Y␣␣Z
Basis␣set
```

```
C.ANO-L...4s3p2d1f.
C␣␣␣␣␣␣␣␣␣.00000000␣␣␣␣.00000000␣␣␣␣␣1.4
End␣of␣basis
End␣of␣input
␣&SCF␣&END
Title
␣C2
ITERATIONS
␣40
Occupied
␣␣2␣␣1␣␣1␣␣0␣␣2␣␣0␣␣0␣␣0
End␣of␣input
␣&RASSCF␣&END
Title
␣C2
Nactel
␣␣4␣␣0␣␣0
Spin
␣␣1
Symmetry
␣␣1
Inactive
␣␣2␣␣0␣␣0␣␣0␣␣2␣␣0␣␣0␣␣0
Ras2
␣␣1␣␣1␣␣1␣␣0␣␣1␣␣1␣␣1␣␣0
*Average
*2␣2␣3␣6␣7
Supsymmetry
1
␣3␣6␣9␣11
1
␣1␣6
1
␣1␣6
0
1
␣3␣5␣8␣12
1
␣1␣6
1
␣1␣6
0
Iter
50,25
Lumorb
End␣of␣input
```

In this case the SCF orbitals are already clean symmetry adapted orbitals (within the printed accuracy). We can then directly use the SUPSYM option. In symmetries a_g and b_{1u} we restrict the rotations among the σ and the δ orbitals, and in symmetries b_{3u}, b_{2u}, b_{2g}, and b_{3g} the rotations among π and ϕ orbitals. Additionally, symmetries b_{3u} and b_{2u} and symmetries b_{2g} and b_{3g} are averaged, respectively, by using the AVERAGE option. They belong to the Π_u and Π_g representations in $D_{\infty h}$, respectively.

A detailed explanation on different CASSCF calculations on the C_2 molecule and their states can be found elsewhere [21]. Instead we include here an example of how to combine the use of UNIX shell script commands with *MOLCAS* as a powerful tool.

The following example computes the transition dipole moment for the transition from the $^1\Sigma_g^+$ state to the $^1\Pi_u$ state in the C_2 molecule. This transition is known as the Phillips bands [24]. This is not a serious attempt to compute this property accurately, but serves as an example of how to set up an automatic calculation. The potential curves are computed using CASSCF wavefunctions along with the transition dipole moment.

Starting orbitals are generated by computing a CI wavefunction once and using the natural orbitals. We loop over a set of distances, compute the CASSCF wave functions for both states and use `RASSI` to compute the TDMs. Several UNIX commands are used to manipulate input and output files, such as grep, sed, and the *awk* language. For instance, an explicit 'sed' is used to insert the geometry into the seward input; the final CASSCF energy is extracted with an explicit 'grep', and the TDM is extracted from the RASSI output using an *awk* script. We are not going to include the *awk* scripts here. Other tools can be used to obtain and collect the data.

In the first script, when the loop over geometries is done, four files are available: geom.list (contains the distances), tdm.list (contains the TDMs), e1.list (contains the energy for the $^1\Sigma_g^+$ state), and e2.list (contains the energy for the $^1\Pi_u$ state). In the second script the vibrational wave functions for the two states and the vibrationally averaged TDMs are now computed using the `VIBROT` program. We will retain the RASSCF outputs in the scratch directory to check the wave function. It is always dangerous to assume that the wave functions will be correct in a CASSCF calculation. Different problems such as root flippings or incorrect orbitals rotating into the active space are not uncommon. Also, it is always necessary to control that the CASSCF calculation has converged. The first script (Korn shell) is:

```
#!/bin/ksh
#
# perform some initializations
#
export Project='C2'
export WorkDir=/temp/$LOGNAME/$Project
export Home=/u/$LOGNAME/$Project
echo "No log" > current.log
trap 'cat current.log ; exit 1' ERR
mkdir $WorkDir
cd $WorkDir
#
# Loop over the geometries and generate input for vibrot
#
list="1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 5.0 10.0"
scf='yes'
print "Sigma" > e1.list
print "Pi" > e2.list
for geom in $list
do
   #--- run seward
   print "Dist $geom" >> geom.list
   sed -e "s/#/$geom/" $Home/$Project.seward.input > seward.input
   molcas seward.input > current.log
   #--- optionally run scf, motra, guga and mrci to obtain good starting orbitals
   if [ "$scf" = 'yes' ]
   then
      scf='no'
      molcas    $Home/$Project.scf.input > current.log
      molcas    $Home/$Project.motra.input > current.log
      molcas    $Home/$Project.guga.input > current.log
      molcas    $Home/$Project.mrci.input > current.log
      cp $Project.CiOrb $Project.RasOrb1
      cp $Project.CiOrb $Project.RasOrb2
   fi
   #--- rasscf wavefunction for 1Sg+
   ln -fs $Project.Job001 JOBIPH
   ln -fs $Project.RasOrb1 INPORB
   molcas   $Home/$Project.rasscf1.input > current.log
   cat current.log >> rasscf1.log
```

```
   cat current.log | grep -i 'average ci' >> e1.list
   cp $Project.RasOrb $Project.RasOrb1
   rm -f JOBIPH INPORB
   #--- rasscf wavefunction for 1Pu
   ln -fs $Project.Job002 JOBIPH
   ln -fs $Project.RasOrb2 INPORB
   molcas   $Home/$Project.rasscf2.input > current.log
   cat current.log >> rasscf2.log
   cat current.log | grep -i 'average ci' >> e2.list
   cp $Project.RasOrb $Project.RasOrb2
   rm -f JOBIPH INPORB
   #--- rassi to obtain transition
   ln -fs $Project.Job001 JOB001
   ln -fs $Project.Job002 JOB002
   molcas   $Home/$Project.rassi.input > current.log
   awk -f $Home/tdm.awk current.log >> tdm.list
   rm -f JOB001 JOB002
   #---
done
#
# Finished so clean up the files.
#
print "Calculation finished" >&2
cd -
rm $WorkDir/molcas.temp*
#rm -r $WorkDir
exit 0
```

In a second script we will compute the vibrational wave functions

```
#!/bin/ksh
#
# perform some initializations
#
export Project='C2'
export WorkDir=/temp/$LOGNAME/$Project
export Home=/u/$LOGNAME/$Project
echo "No log" > current.log
trap 'cat current.log ; exit 1' ERR
mkdir $WorkDir
cd $WorkDir
#
# Build vibrot input
#
cp e1.list $Home
cp e2.list $Home
cp geom.list $Home
cp tdm.list $Home
#---
cat e1.list geom.list | awk -f $Home/wfn.awk > vibrot1.input
cat e2.list geom.list | awk -f $Home/wfn.awk > vibrot2.input
cat tdm.list geom.list | awk -f $Home/tmc.awk > vibrot3.input
#---
ln -fs $Project.VibWvs1 VIBWVS
molcas  vibrot1.input > current.log
cat current.log
rm -f VIBWVS
#---
ln -fs $Project.VibWvs2 VIBWVS
molcas  vibrot2.input > current.log
cat current.log
rm -f VIBWVS
#---
ln -fs $Project.VibWvs1 VIBWVS1
ln -fs $Project.VibWvs2 VIBWVS2
molcas  vibrot3.input > current.log
```

```
cat current.log
rm -f VIBWVS1 VIBWVS2
#
# Finished so clean up the files.
#
print "Calculation finished" >&2
cd -
rm $WorkDir/molcas.temp*
#rm -r $WorkDir
exit 0
```

The input for the first part of the calculations include the SEWARD, SCF, MOTRA, GUGA, and MRCI inputs:

```
␣&SEWARD␣&END
Title
␣C2
Pkthre
1.0D-11
Symmetry
␣␣X␣␣Y␣␣Z
Basis␣set
C.ANO-S...3s2p.
C␣␣␣␣␣␣␣␣.00000000␣␣␣␣.00000000␣␣␣#
End␣of␣basis
End␣of␣input
␣&SCF␣&END
Title
␣C2
ITERATIONS
␣40
Occupied
␣␣2␣␣1␣␣1␣␣0␣␣2␣␣0␣␣0␣␣0
End␣of␣input
␣&MOTRA␣&END
Title
␣C2␣molecule
Frozen
␣1␣0␣0␣0␣1␣0␣0␣0
LumOrb
End␣of␣input
␣&GUGA␣&END
Title
␣C2␣molecule
Electrons
␣␣␣␣8
Spin
␣␣␣␣1
Inactive
␣␣␣␣1␣␣␣␣1␣␣␣␣1␣␣␣␣0␣␣␣␣1␣␣␣␣0␣␣␣␣0␣␣␣␣0
Active
␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0
CiAll
␣␣␣␣1
End␣of␣Input
␣&MRCI␣&END
Title
␣C2␣molecule
SDCI
End␣of␣input
```

We are going to use a small ANO [$3s2p$] basis set because our purpose it is not to obtain an extreme accuracy. In the SEWARD input the sign '#' will be replaced by the right distance using the 'sed' command. In the MOTRA input we have frozen the two core orbitals in the molecule, which will be recognized by the MRCI program. The GUGA input defines the reference space of configurations for the subsequent MRCI or ACPF calculation. In this case the valence orbitals are doubly occupied and there is only one reference configuration (they are included as inactive). We thus use one single configuration to perform the SDCI calculation and obtain the initial set of orbitals for the CASSCF calculation.

The lowest $^1\Sigma_g^+$ state in C_2 is the result of the electronic configuration [core]$(2\sigma_g)^2$ $(2\sigma_u)^2$ $(1\pi_u)^4$. Only one electronic state is obtained from this configuration. The configuration $(1\pi_u)^3$ $(3\sigma_g)^1$ is close in energy and generates two possibilities, one $^3\Pi_u$ and one $^1\Pi_u$ state. The former is the lowest state of the Swan bands, and was thought to be the ground state of the molecule. Transitions to the $^1\Pi_u$ state are known as the Phillips band and this is the state we are going to compute. We have the possibility to compute the state in symmetry b_{3u} or b_{2u} (MOLCAS symmetry groups 2 and 3, respectively) in the D_{2h} group, because both represent the degenerate Π_u symmetry in $D_{\infty h}$.

The RASSCF input file to compute the two states are:

```
␣&RASSCF␣&END
Title
␣C2␣1Sigmag+␣state.
Nactel
␣␣4␣␣0␣␣0
Spin
␣␣1
Symmetry
␣␣1
Inactive
␣␣2␣␣0␣␣0␣␣0␣␣2␣␣0␣␣0␣␣0
Ras2
␣␣1␣␣1␣␣1␣␣0␣␣1␣␣1␣␣1␣␣0
*Average
*2␣2␣3␣6␣7
OutOrbitals
␣Natural
␣1
Iter
50,25
Lumorb
End␣of␣input
```

```
␣&RASSCF␣&END
Title
␣C2␣1Piu␣state.
Nactel
␣␣4␣␣0␣␣0
Spin
␣␣1
Symmetry
␣␣2
Inactive
␣␣2␣␣0␣␣0␣␣0␣␣2␣␣0␣␣0␣␣0
Ras2
␣␣1␣␣1␣␣1␣␣0␣␣1␣␣1␣␣1␣␣0
Average
2␣2␣3␣6␣7
OutOrbitals
␣Natural
␣1
Iter
50,25
Lumorb
```

```
End␣of␣input
```

We can skip the SUPSYM option because our basis set contains only s, p functions and no undesired rotations can happen. Symmetries b_{3u} and b_{2u} on one hand and b_{2g} and b_{3g} on the other are averaged. Notice that to obtain natural orbitals we have used keyword OUTORBITALS instead of the old `RASREAD` program. In addition, we need the `RASSI` input:

```
␣&RASSI␣&END
NrOfJobiphs
␣2␣1␣1
␣1
␣1
End␣of␣input
```

The VIBROT inputs to compute the vibrational-rotational analysis and spectroscopic constants of the state should be:

```
␣&VIBROT␣&END
RoVibrational␣spectrum
Title
␣Vib-Rot␣spectrum␣for␣C2.␣1Sigmag+
Atoms
0␣C␣0␣C
Grid
400
Range
2.0␣10.0
Vibrations
3
Rotations
0␣4
Orbital
0
Potential
2.2␣-75.42310136
...
End␣of␣input
```

Under the keyword POTENTIAL the bond distance and potential energy (both in au) of the corresponding state must be included. In this case we are going to compute three vibrational quanta and four rotational quantum numbers. For the $^1\Pi_u$ state, the keyword ORBITAL must be set to one, corresponding to the orbital angular momentum of the computed state. `VIBROT` fits the potential curve to an analytical curve using splines. The ro-vibrational Schrödinger equation is then solved numerically (using Numerov's method) for one vibrational state at a time and for the specified number of rotational quantum numbers. File *`VIBWVS`* will contain the corresponding wave function for further use.

Just to give some of the results obtained, the spectroscopic constants for the $^1\Sigma_g^+$ state were:

```
      Re(a)                1.4461
      De(ev)               3.1088
      D0(ev)               3.0305
      we(cm-1)         .126981E+04
      wexe(cm-1)      -.130944E+02
      weye(cm-1)      -.105159E+01
      Be(cm-1)         .134383E+01
      Alphae(cm-1)     .172923E-01
      Gammae(cm-1)     .102756E-02
      Dele(cm-1)       .583528E-05
```

```
Betae(cm-1)      .474317E-06
```

and for the $^1\Pi_u$ state:

```
Re(a)                 1.3683
De(ev)                2.6829
D0(ev)                2.5980
we(cm-1)         .137586E+04
wexe(cm-1)      -.144287E+02
weye(cm-1)       .292996E+01
Be(cm-1)         .149777E+01
Alphae(cm-1)     .328764E-01
Gammae(cm-1)     .186996E-02
Dele(cm-1)       .687090E-05
Betae(cm-1)     -.259311E-06
```

To compute vibrationally averaged TDMs the `VIBROT` input must be:

```
␣&VIBROT␣&END
Transition␣moments
Observable
Transition␣dipole␣moment
2.2␣0.412805
...
End␣of␣input
```

Keyword OBSERVABLE indicates the start of input for radial functions of observables other than the energy. In the present case the vibrational-rotational matrix elements of the transition dipole moment function will be generated. The values of the bond distance and the TDM at each distance must be then included in the input. VIBROT also requires the *`VIBWVS1`* and *`VIBWVS2`* files containing the vibrational wave functions of the involved electronic states. The results obtained contain matrix elements, transition moments over vibrational wave functions, and the lifetimes of the transition among all the computed vibrational-rotational states. The radiative lifetime of a vibrational level depends on the sum of the transition probabilities to all lower vibrational levels in all lower electronic states. If rotational effects are neglected, the lifetime (τ'_v) can be written as

$$\tau'_v = (\sum_{v''} A_{v'v''})^{-1} \tag{5.1}$$

where v' and v'' are the vibrational levels of the lower and upper electronic state and $A_{v'v''}$ is the Einstein A coefficient (ns^{-1}) computed as

$$A_{v'v''} = 21.419474\ (\Delta E_{v'v''})^3 (TDM_{v'v''})^2 \tag{5.2}$$

$\Delta E_{v'v''}$ is the energy difference (au) and $TDM_{v'v''}$ the transition dipole moment (au) of the transition.

For instance, for rotational states zero of the $^1\Sigma_g^+$ state and one of the $^1\Pi_u$ state:

```
Rotational quantum number for state 1:  0, for state 2:  1
-----------------------------------------------------------------------------

Overlap matrix for vibrational wave functions for state number  1
1  1  .307535  2  1  .000000  2  2  .425936  3  1  .000000  3  2  .000000  3  3  .485199
```

```
  Overlap matrix for vibrational wave functions for state number  2
  1  1  .279631  2  1  .000000  2  2  .377566  3  1  .000000  3  2  .000000  3  3  .429572

  Overlap matrix for state 1 and state 2 functions
    -.731192  -.617781  -.280533
     .547717  -.304345  -.650599
    -.342048   .502089  -.048727

  Transition moments over vibrational wave functions (atomic units)
    -.286286  -.236123  -.085294
     .218633  -.096088  -.240856
    -.125949   .183429   .005284

  Energy differences for vibrational wave functions(atomic units)
  1  1  .015897  2  1  .010246  2  2  .016427  3  1  .004758  3  2  .010939  3  3  .017108

  Contributions to inverse lifetimes (ns-1)
 No degeneracy factor is included in these values.
  1  1  .000007  2  1  .000001  2  2  .000001  3  1  .000000  3  2 .000001   3  3  .000000

  Lifetimes (in nano seconds)
    v       tau
    1 122090.44
    2  68160.26
    3  56017.08
```

Probably the most important caution when using the VIBROT program in diatomic molecules is that the number of vibrational states to compute and the accuracy obtained depends strongly on the computed surface. In the present case we compute all the curves to the dissociation limit. In other cases, the program will complain if we try to compute states which lie at energies above those obtained in the calculation of the curve.

5.1.3 A transition metal dimer: Ni_2

This section is a brief comment on a complex situation in a diatomic molecule such as Ni_2. Our purpose is to compute the ground state of this molecule. An explanation of how to calculate it accurately can be found in ref. [22]. However we will concentrate on computing the electronic states at the CASSCF level.

The nickel atom has two close low-lying configurations $3d^84s^2$ and $3d^94s^1$. The combination of two neutral Ni atoms leads to a Ni_2 dimer whose ground state has been somewhat controversial. For our purposes we commence with the assumption that it is one of the states derived from $3d^94s^1$ Ni atoms, with a single bond between the $4s$ orbitals, little $3d$ involvement, and the holes localized in the $3d\delta$ orbitals. Therefore, we compute the states resulting from two holes on δ orbitals: $\delta\delta$ states.

We shall not go through the procedure leading to the different electronic states that can arise from these electronic configurations, but refer to the Herzberg book on diatomic molecules [24] for details. In $D_{\infty h}$ we have three possible configurations with two holes, since the δ orbitals can be either *gerade* (g) or *ungerade* (u): $(\delta_g)^{-2}$, $(\delta_g)^{-1}(\delta_u)^{-1}$, or $(\delta_u)^{-2}$. The latter situation corresponds to nonequivalent electrons while the other two to equivalent electrons. Carrying through the analysis we obtain the following electronic states:

$(\delta_g)^{-2}$: ${}^1\Gamma_g$, ${}^3\Sigma_g^-$, ${}^1\Sigma_g^+$
$(\delta_u)^{-2}$: ${}^1\Gamma_g$, ${}^3\Sigma_g^-$, ${}^1\Sigma_g^+$
$(\delta_g)^{-1}(\delta_u)^{-1}$: ${}^3\Gamma_u$, ${}^1\Gamma_u$, ${}^3\Sigma_u^+$, ${}^3\Sigma_u^-$, ${}^1\Sigma_u^+$, ${}^1\Sigma_u^-$

In all there are thus 12 different electronic states.

Next, we need to classify these electronic states in the lower symmetry D_{2h}, in which MOLCAS works. This is done in Table 5.7, which relates the symmetry in $D_{\infty h}$ to that of D_{2h}. Since we have only Σ^+, Σ^-, and Γ states here, the D_{2h} symmetries will be only A_g, A_u, B_{1g}, and B_{1u}. The table above can now be rewritten in D_{2h}:

$(\delta_g)^{-2}$: $({}^1\mathrm{A}_g + {}^1\mathrm{B}_{1g})$, ${}^3\mathrm{B}_{1g}$, ${}^1\mathrm{A}_g$
$(\delta_u)^{-2}$: $({}^1\mathrm{A}_g + {}^1\mathrm{B}_{1g})$, ${}^3\mathrm{B}_{1g}$, ${}^1\mathrm{A}_g$
$(\delta_g)^{-1}(\delta_u)^{-1}$: $({}^3\mathrm{A}_u + {}^3\mathrm{B}_{1u})$, $({}^1\mathrm{A}_u + {}^1\mathrm{B}_{1u})$, ${}^3\mathrm{B}_{1u}$, ${}^3\mathrm{A}_u$, ${}^1\mathrm{B}_{1u}$, ${}^1\mathrm{A}_u$

or, if we rearrange the table after the D_{2h} symmetries:

${}^1\mathrm{A}_g$: ${}^1\Gamma_g(\delta_g)^{-2}$, ${}^1\Gamma_g(\delta_u)^{-2}$, ${}^1\Sigma_g^+(\delta_g)^{-2}$, ${}^1\Sigma_g^+(\delta_u)^{-2}$
${}^1\mathrm{B}_{1u}$: ${}^1\Gamma_u(\delta_g)^{-1}(\delta_u)^{-1}$, ${}^1\Sigma_u^+(\delta_g)^{-1}(\delta_u)^{-1}$
${}^1\mathrm{B}_{1g}$: ${}^1\Gamma_g(\delta_g)^{-2}$, ${}^1\Gamma_g(\delta_u)^{-2}$
${}^1\mathrm{A}_u$: ${}^1\Gamma_u(\delta_g)^{-1}(\delta_u)^{-1}$, ${}^1\Sigma_u^-(\delta_g)^{-1}(\delta_u)^{-1}$

${}^3\mathrm{B}_{1u}$: ${}^3\Gamma_u(\delta_g)^{-1}(\delta_u)^{-1}$, ${}^3\Sigma_u^+(\delta_g)^{-1}(\delta_u)^{-1}$
${}^3\mathrm{B}_{1g}$: ${}^3\Sigma_g^-(\delta_g)^{-2}$, ${}^3\Sigma_g^-(\delta_u)^{-2}$
${}^3\mathrm{A}_u$: ${}^3\Gamma_u(\delta_g)^{-1}(\delta_u)^{-1}$, ${}^3\Sigma_u^-(\delta_g)^{-1}(\delta_u)^{-1}$

It is not necessary to compute all the states because some of them (the Γ states) have degenerate components. It is both possible to make single state calculations looking for the lowest energy state of each symmetry or state-average calculations in each of the symmetries. The identification of the $D_{\infty h}$ states can be somewhat difficult. For instance, once we have computed one ${}^1\mathrm{A}_g$ state it can be a ${}^1\Gamma_g$ or a ${}^1\Sigma_g^+$ state. In this case the simplest solution is to compare the obtained energy to that of the ${}^1\Gamma_g$ degenerate component in B_{1g} symmetry, which must be equal to the energy of the ${}^1\Gamma_g$ state computed in A_g symmetry. Other situations can be more complicated and require a detailed analysis of the wave function.

It is important to have clean *d*-orbitals and the SUPSym keyword may be needed to separate δ and σ (and γ if g-type functions are used in the basis set) orbitals in symmetry 1 (A_g). The AVERage keyword is not needed here because the π and ϕ orbitals have the same occupation for Σ and Γ states.

Finally, when states of different multiplicities are close in energy, the spin-orbit coupling which mix the different states should be included. The CASPT2 study of the Ni_2 molecule in reference [22], after considering all the mentioned effects determined that the ground state of the molecule is a 0_g^+ state, a mixture of the ${}^1\Sigma_g^+$ and ${}^3\Sigma_g^-$ electronic states. For a review of the spin-orbit coupling and other important coupling effects see reference [27].

5.1.4 High symmetry systems in MOLCAS

There are a large number of symmetry point groups in which MOLCAS cannot directly work. Although unusual in organic chemistry, some of them can be easily found in inorganic compounds. Systems belonging for instance to three-fold groups such as C_{3v}, D_{3h}, or D_{6h}, or to groups such O_h or D_{4h} must be computed using lower symmetry point groups. The consequence is, as in linear molecules, that orbitals and states belonging to different representations in the actual groups, belong to the same representation in the lower symmetry case, and *vice versa*. In the `RASSCF` program it is possible to prevent the orbital and configurational mixing caused by the first situation. The CLEAnup and SUPSymmetry keywords can be used in a careful, and somewhat tedious, way. The right symmetry behaviour of the RASSCF wave function is then assured. It is sometimes not a trivial task to identify the symmetry of the orbitals in the higher symmetry representation and which coefficients must vanish. In many situations the ground state wave function keeps the rigth symmetry (at least within the printing accuracy) and helps to identify the orbitals and coefficients. It is more frequent that the mixing happens for excited states.

The reverse situation, that is, that orbitals (normally degenerated) which belong to the same symmetry representation in the higher symmetry groups belong to different representations in the lower symmetry groups cannot be solved by the present implementation of the `RASSCF` program. The AVERage keyword, which performs this task in the linear molecules, is not prepared to do the same in non-linear systems. Provided that the symmetry problems mentioned in the previous paragraph are treated in the proper way and the trial orbitals have the right symmetry, the `RASSCF` code behaves properly.

There is a important final precaution concerning the high symmetry systems: the geometry of the molecule must be of the right symmetry. Any deviation will cause severe mixings. Figure 5.1 contains the `SEWARD` input for the magnesium porphirin molecule. This is a D_{4h} system which must be computed D_{2h} in MOLCAS.

For instance, the x and y coordinates of atoms C1 and C5 are interchanged with equal values in D_{4h} symmetry. Both atoms must appear in the `SEWARD` input because they are not independent by symmetry in the D_{2h} symmetry in which MOLCAS is going to work. Any deviation of the values, for instance to put the y coordinate to 0.681879 Å in C1 and the x to 0.681816 Å in C5 and similar deviations for the other coordinates, will lead to severe symmetry mixtures. This must be taken into account when geometry data are obtained from other program outputs or data bases.

Figure 5.1: Sample input of the SEWARD program for the magnesium porphirin molecule in the D_{2h} symmetry

```
␣&SEWARD␣&END
Title
␣Mg-Porphyrine␣D4h␣computed␣D2h
Symmetry
␣X␣Y␣Z
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣4.254984␣␣␣␣␣.681879␣␣␣␣␣.000000␣␣Angstrom
C2␣␣␣␣2.873412␣␣␣␣1.101185␣␣␣␣0.000000␣␣Angstrom
C3␣␣␣␣2.426979␣␣␣␣2.426979␣␣␣␣0.000000␣␣Angstrom
C4␣␣␣␣1.101185␣␣␣␣2.873412␣␣␣␣0.000000␣␣Angstrom
C5␣␣␣␣␣.681879␣␣␣␣4.254984␣␣␣␣0.000000␣␣Angstrom
End␣of␣basis
Basis␣set
N.ANO-S...3s2p1d.
N1␣␣␣␣2.061400␣␣␣␣␣.000000␣␣␣␣0.000000␣␣Angstrom
N2␣␣␣␣␣.000000␣␣␣␣2.061400␣␣␣␣0.000000␣␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-S...2s0p.
H1␣␣␣␣5.109145␣␣␣␣1.348335␣␣␣␣0.000000␣␣Angstrom
H3␣␣␣␣3.195605␣␣␣␣3.195605␣␣␣␣0.000000␣␣Angstrom
H5␣␣␣␣1.348335␣␣␣␣5.109145␣␣␣␣0.000000␣␣Angstrom
End␣of␣basis
Basis␣set
Mg.ANO-S...4s3p1d.
```

```
Mg␣␣␣␣␣.000000␣␣␣␣␣.000000␣␣␣␣0.000000␣␣Angstrom
End␣of␣basis
End␣of␣Input
```

The situation can be more complex for some three-fold point groups such as D_{3h} or C_{3v}. In these cases it is not possible to input in the exact cartesian geometry, which depends on trigonometric relations and relies on the numerical precision of the coordinates entry. It is necessary then to use in the `SEWARD` input as much precision as possible and check on the distance matrix of the `SEWARD` output if the symmetry of the system has been kept at least within the output printing criteria.

5.2 Geometry optimizations and Hessians.

To optimize a molecular geometry is probably one of the most frequent interests of a quantum chemist [28]. In the present section we examine some examples of obtaining stationary points on the energy surfaces. We will focus in this section in searching of minimal energy points, postponing the discussion on transition states to section 5.3. This type of calculations require the computation of molecular gradients, whether using analytical or numerical derivatives. We will also examine how to obtain the full geometrical Hessian for a molecular state, what will provide us with vibrational frequencies within the harmonic approximation and thermodynamic properties by the use of the proper partition functions.

The program `ALASKA` computes analytical gradients for optimized wave functions. In 7.4 the SCF, DFT, and CASSCF/RASSCF levels of calculation are available. The program `ALASKA` also computes numerical gradients from CASPT2 and MS-CASPT2 energies. Provided with the first order derivative matrix with respect to the nuclei and an approximate guess of the Hessian matrix, the program SLAPAF is then used to optimize molecular structures. From MOLCAS-5 it is not necessary to explicitly define the set of internal coordinates of the molecule in the SLAPAF input. Instead a redundant coordinates approach is used. If the definition is absent the program builds its own set of parameters based on curvature-weighted non-redundant internal coordinates and displacements [29]. As they depend on the symmetry of the system it might be somewhat difficult in some systems to define them. It is, therefore, strongly recommended to let the program define its own set of non-redundant internal coordinates. In certain situations such as bond dissociations the previous coordinates may not be appropriate and the code directs the user to use instead Cartesian coordinates, for instance.

5.2.1 Ground state optimizations and vibrational analysis

As an example we are going to work with the 1,3-cyclopentadiene molecule. This is a five-carbon system forming a ring which has two conjugated double bonds. Each carbon has one attached hydrogen atom except one which has two. We will use the CASSCF method and take advantage of the symmetry properties of the molecule to compute ground and excited states. To ensure the convergence of the results we will also perform Hessian calculations to compute the force fields at the optimized geometries.

In this section we will combine two types of procedures to perform calculations in MOLCAS. The user may then choose the most convenient for her/his taste. We can use an general script and perform an input-oriented calculation, when all the information relative to the calculation, including links for the files and control of iterations, are inserted in the input file. The other procedure is the classical script-oriented system used in previous examples and typically previous versions of MOLCAS. Let's start by making an input-oriented optimization. A script is still needed to perform the basic definitions, although they can be mostly done within the input file. A suggested form for this general script could be:

```
#!/bin/sh
export MOLCAS=/home/molcas/molcashome
export MOLCASMEM=64
export Project=Cyclopentadiene1
export HomeDir=/home/somebody/somewhere
export WorkDir=$HomeDir/$Project
[ ! -d $WorkDir ] && mkdir $WorkDir
molcas $HomeDir/$Project.input >$HomeDir/$Project.out 2>$HomeDir/$Project.err
exit
```

We begin by defining the input for the initial calculation. In simple cases the optimization procedure is very efficient. We are going, however, to design a more complete procedure that may help in more complex situations. It is sometimes useful to start the optimization in a small size basis set and use the obtained approximate Hessian to continue the calculation with larger basis sets. Therefore, we will begin by using the minimal STO-3G basis set to optimize the ground state of 1,3-cyclopentadiene within C_{2v} symmetry.

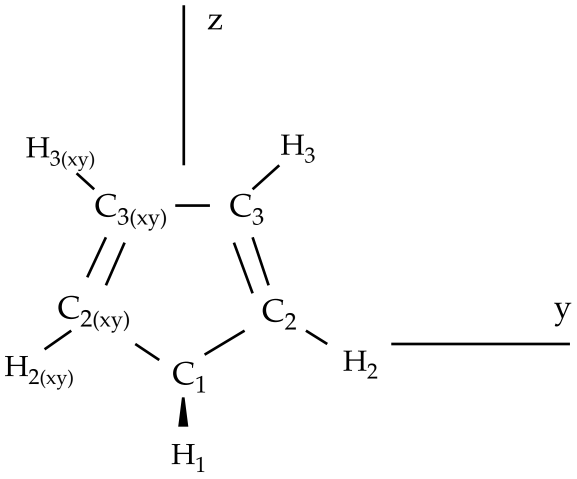


Figure 5.2: 1,3-cyclopentadiene

We will use the following input in an input-oriented calculation. Notice that we have directed the output files sequentially (one per iteration) to the $WorkDir directory by using the Set Output File command, the maximum number of iterations of the subsequent loops, and the starting and end of the loops on each step of the optimization procedure by using the commands Do while and EndDo. It is important than the parameter MaxIter never goes beyond the number of iterations in the SLAPAF input.

```
>>>␣Set␣Output␣File␣<<<
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
!ln␣-fs␣$HomeDir/$Project.ForceConstant.STO-3G␣RUNFILE
␣&SEWARD␣&END
Title
1,3,-cyclopentadiene.␣STO-3G␣basis␣set.
```

```
Symmetry
␣X␣XY
Basis␣set
C.STO-3G....
C1␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣␣Bohr
C2␣␣␣␣0.000000␣␣2.222644␣␣1.774314␣␣Bohr
C3␣␣␣␣0.000000␣␣1.384460␣␣4.167793␣␣Bohr
End␣of␣basis
Basis␣set
H.STO-3G....
H1␣␣␣␣1.662033␣␣0.000000␣-1.245623␣␣Bohr
H2␣␣␣␣0.000000␣␣4.167844␣␣1.149778␣␣Bohr
H3␣␣␣␣0.000000␣␣2.548637␣␣5.849078␣␣Bohr
End␣of␣basis
End␣of␣Input
␣&SCF␣&END
TITLE
␣cyclopentadiene␣molecule
OCCUPIED
9␣1␣6␣2
ITERATIONS
40
END␣OF␣INPUT
␣&RASSCF␣&END
TITLE
␣cyclopentadiene␣molecule␣1A1
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣␣9␣␣␣␣0␣␣␣␣6␣␣␣␣0
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3␣␣␣␣␣␣␣␣␣␣␣␣<---␣All␣pi␣valence␣orbitals␣active
ITER
50,25
CIMX
25
LUMORB
END␣OF␣INPUT
␣&ALASKA␣&END
End␣of␣Input
␣&SLAPAF␣&END
Iterations
80
Thrs
0.5D-06␣1.0D-03
End␣of␣Input
>>>␣EndDo␣<<<
```

A link to the ***RUNFILE*** file has been made within the input stream. This saves the file for use as a guess of the Hessian matrix in the following calculation. The link can be also done in the shell script.

The generators used to define the C_{2v} symmetry are X and XY, plane yz and axis z. They differ from those used in other examples as in section 5.1.1. The only consequence is that the order of the symmetries in `SEWARD` differs. In the present case the order is: a_1, a_2, b_1, and b_2, and consequently the classification by symmetries of the orbitals in the SCF and RASSCF inputs will differ. It is therefore recommended to initially use the option TEST in the `SEWARD` input to check the symmetry option. This option, however, will stop the calculation after the `SEWARD` input head is printed.

The calculation converges in three steps. We change now the input. We can choose between replacing by hand the geometry of the `SEWARD` input or use the same $WorkDir directory and let the program to take the last geometry stored into the communication ***RUNFILE*** file. In any case the new input can be:

```
>>>␣Set␣Output␣File␣<<<
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
!ln␣-fs␣$HomeDir/$Project.ForceConstant.STO-3G␣COMOLD
␣&SEWARD␣&END
Title
1,3,-cyclopentadiene␣molecule
Symmetry
␣X␣XY
Basis␣set
C.ANO-L...4s3p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-2.3726116671
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.2447443782␣␣␣␣␣␣␣␣-.5623842095
C3␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣1.4008186026␣␣␣␣␣␣␣␣1.8537195887
End␣of␣basis
Basis␣set
H.ANO-L...2s.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣1.6523486260␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-3.6022531906
H2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣4.1872267035␣␣␣␣␣␣␣-1.1903003793
H3␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.5490335048␣␣␣␣␣␣␣␣3.5419847446
End␣of␣basis
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<<
␣&SCF␣&END
TITLE
␣cyclopentadiene␣molecule
OCCUPIED
9␣1␣6␣2
ITERATIONS
40
END␣OF␣INPUT
␣&RASSCF␣&END
LUMORB
TITLE
␣cyclopentadiene␣molecule␣1A1
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣␣9␣␣␣␣0␣␣␣␣6␣␣␣␣0
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
ITER
50,25
CIMX
25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIREstart
TITLE
␣cyclopentadiene␣molecule␣1A1
SYMMETRY
␣␣␣␣1
SPIN
```

```
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣␣9␣␣␣␣0␣␣␣␣6␣␣␣␣0
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
ITER
50,25
CIMX
25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
␣&ALASKA␣&END
End␣of␣file
␣&SLAPAF␣&END
OldForce␣Constant␣Matrix
Iterations
80
Thrs
0.5D-06␣1.0D-03
End␣of␣Input
>>>␣EndDo␣<<<
```

The ***RUNOLD*** file will be used by SLAPAF as initial Hessian to carry out the relaxation. This use of the ***RUNFILE*** can be done between any different calculations provided they work in the same symmetry.

In the new basis set, the resulting optimized geometry at the CASSCF level in C_{2v} symmetry is:

```
********************************************
* Values of internal coordinates           *
********************************************
C2C1   2.851490 Bohr
C3C2   2.545737 Bohr
C3C3   2.790329 Bohr
H1C1   2.064352 Bohr
H2C2   2.031679 Bohr
H3C3   2.032530 Bohr
C1C2C3     109.71 Degrees
C1C2H2     123.72 Degrees
C2C3H3     126.36 Degrees
H1C1H1     107.05 Degrees
```

Once we have the optimized geometry we can obtain the force field, to compute the force constant matrix and obtain an analysis of the harmonic frequency. This is done by computing the analytical Hessian at the optimized geometry. Notice that this is a single-shot calculation using the `MCKINLEY`, which will automatically start the `MCLR` module in case of a frequency calculation.

```
␣&SEWARD␣&END
Title
1,3,-cyclopentadiene␣molecule
Symmetry
␣X␣XY
Basis␣set
C.ANO-L...4s3p1d.
␣␣␣␣␣C1␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-2.3483061484
␣␣␣␣␣C2␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.2245383122␣␣␣␣␣␣␣-0.5643712787
␣␣␣␣␣C3␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣1.3951643642␣␣␣␣␣␣␣␣1.8424767578
End␣of␣basis
```

```
Basis␣set
H.ANO-L...2s.
␣␣␣␣␣H1␣␣␣␣␣␣␣␣␣␣␣␣␣1.6599988023␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-3.5754797471
␣␣␣␣␣H2␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣4.1615845660␣␣␣␣␣␣␣-1.1772096132
␣␣␣␣␣H3␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.5501642966␣␣␣␣␣␣␣␣3.5149458446
End␣of␣basis
End␣of␣Input
␣&SCF␣&END
TITLE
␣cyclopentadiene␣molecule
OCCUPIED
9␣1␣6␣2
ITERATIONS
40
END␣OF␣INPUT
␣&RASSCF␣&END
TITLE
␣cyclopentadiene␣molecule␣1A1
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣␣9␣␣␣␣0␣␣␣␣6␣␣␣␣0
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
ITER
50,25
CIMX
25
LUMORB
END␣OF␣INPUT
␣&MCKINLEY␣&END
Perturbation
␣Hessian
End␣of␣Input
```

Cyclopentadiene has 11 atoms, that mean 3N = 33 Cartesian degrees of freedom. Therefore the `MCLR` output will contain 33 frequencies. From those, we are just interested in the 3N-6 = 27 final degrees of freedom that correspond to the normal modes of the system. We will discard from the output the three translational (T_i) and three rotational (R_i) coordinates. The table of characters gives us the classification of these six coordinates: a_1 (T_z), a_2 (R_z), b_2 (T_x,R_y), b_1 (T_y,R_x). This information is found in the Seward output:

```
Character Table for C2v
         E   s(yz) C2(z) s(xz)
a1       1     1     1     1  z
a2       1    -1     1    -1  xy, Rz, I
b2       1     1    -1    -1  y, yz, Rx
b1       1    -1    -1     1  x, xz, Ry
```

It is simply to distinguish these frequencies because they must be zero, although and because of numerical un accuracies they will be simply close to zero. In the present calculation the harmonic frequencies, the infrared intensities, and the corresponding normal modes printed below in Cartesian coordinates are the following:

```
    Symmetry a1
   ==============
                         1         2         3         4         5         6
          Freq.       i0.06    847.85    966.07   1044.66   1187.60   1492.41
```

```
      Intensity:   0.340E-08 0.121E-02 0.533E+01 0.415E+00 0.641E-01 0.393E+01
       C1       z    0.12305   0.15517  -0.14426  -0.06780   0.06205   0.02429
       C2       y    0.00000   0.19533   0.13649   0.10357  -0.02549  -0.08379
       C2       z    0.17402  -0.01781   0.06590  -0.00213   0.03194  -0.06400
       C3       y    0.00000   0.02739  -0.06782   0.19744  -0.03160   0.16810
       C3       z    0.17402  -0.09921   0.00301   0.07315  -0.09872  -0.03285
       H1       x    0.00000  -0.01773  -0.00108  -0.00960   0.00375   0.05160
       H1       z    0.17402   0.19434  -0.20604  -0.11326   0.10510   0.12231
       H2       y    0.00000   0.15234   0.26633   0.10257   0.14973   0.08294
       H2       z    0.17402  -0.15622   0.44925  -0.04930   0.60953   0.48030
       H3       y    0.00000  -0.18252  -0.27946   0.49335   0.44520  -0.35864
       H3       z    0.17402   0.04882   0.15083  -0.11220  -0.44201   0.34603
                         7         8         9        10        11
            Freq.    1579.76   1633.39   3140.69   3315.45   3341.28
      Intensity:   0.473E+01 0.432E+00 0.255E+02 0.143E+02 0.571E+01
...
    Symmetry a2
   ==============
                         1         2         3         4         5
            Freq.      i5.81    492.93    663.83    872.55   1235.03
...
    Symmetry b2
   ==============
                         1         2         3         4         5         6
            Freq.     i11.15     i0.05    858.71   1020.49   1173.32   1386.18
      Intensity:   0.249E-01 0.821E-07 0.259E+01 0.743E+01 0.627E-01 0.163E+00
...
                         7         8         9        10
            Freq.    1424.08   1699.08   3305.25   3334.09
      Intensity:   0.966E+00 0.427E+00 0.151E+00 0.302E+02
...
    Symmetry b1
   ==============
                         1         2         3         4         5         6
            Freq.      i8.13      0.08    349.36    663.03    881.26    980.60
      Intensity:   0.463E-01 0.465E-06 0.505E+01 0.896E+02 0.302E+00 0.169E+02
...
                         7
            Freq.    3159.81
      Intensity:   0.149E+02
...
```

Apart from the six mentioned translational and rotational coordinates There are no imaginary frequencies and therefore the geometry corresponds to a stationary point within the C_{2v} symmetry. The frequencies are expressed in reciprocal centimeters.

After the vibrational analysis the zero-point energy correction and the thermal corrections to the total energy, internal, entropy, and Gibbs free energy. The analysis uses the standard expressions for an ideal gas in the canonical ensemble which can be found in any standard statistical mechanics book. The analysis is performed at different temperatures, for instance:

```
␣Temperature␣=␣273.00␣Kelvin
␣---------------------------
␣␣␣␣␣␣␣␣␣␣␣DeltaG␣=␣59.90␣kcal/mol
␣␣␣␣␣␣␣␣␣␣␣␣␣ZPVE␣=␣60.11␣kcal/mol
␣␣␣␣␣␣␣␣TotDeltaU␣=␣60.72␣kcal/mol
␣TotDeltaU␣-␣ZPVE␣=␣␣0.61␣kcal/mol
␣␣␣␣␣␣␣␣␣␣␣DeltaS␣=␣␣3.0001␣eu/mol
␣␣␣␣␣␣␣␣␣␣␣U(T&R)␣=␣␣1.6275␣kcal/mol
␣␣␣␣␣␣␣␣␣Tot(temp)=␣␣2.2393␣kcal/mol
```

Next, polarizabilities (see below) and isotope shifted frequencies are also displayed in the output.

```
␣************************************
␣*␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣*
␣*␣␣␣␣␣␣␣Polarizabilities␣␣␣␣␣␣␣␣␣␣␣*
␣*␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣*
␣************************************
␣␣␣-34.7624440
␣␣␣␣␣0.0000000␣-51.8645093
␣␣␣␣␣0.0000000␣␣␣0.0000000␣-57.7540263
```

For a graphical representation of the harmonic frequencies one can also use the $Project.freq.molden file as an input to the MOLDEN program.

5.2.2 Excited state optimizations

The calculation of excited states using the `ALASKA` and `SLAPAF` codes has no special characteristic. The wave function is defined by the `SCF` or `RASSCF` programs. Therefore if we want to optimize an excited state the `RASSCF` input has to be defined accordingly. It is not, however, an easy task, normally because the excited states have lower symmetry than the ground state and one has to work in low order symmetries if the full optimization is pursued.

Take the example of the thiophene molecule (see fig. 5.10 in next section). The ground state has C_{2v} symmetry: 1 ^{1}A$_1$. The two lowest valence excited states are 2^{1}A$_1$ and 1^{1}B$_2$. If we optimize the geometries within the C_{2v} symmetry the calculations converge easily for the three states. They are the first, second, and first roots of their symmetry, respectively. But if we want to make a full optimization in C$_1$, or even a restricted one in C$_s$, all three states belong to the same symmetry representation. The higher the root more difficult is to converge it. A geometry optimization requires single-root optimized CASSCF wave-functions, but, unlike in previous *MOLCAS* versions, we can now carry out State-Average (SA) CASSCF calculations between different roots. The wave functions we have with this procedure are based on an averaged density matrix, and a further orbital relaxation is required. The `MCLR` program can perform such a task by means of a perturbational approach. Therefore, if we choose to carry out a SA-CASSCF calculations in the optimization procedure, the `ALASKA` module will automatically start up the `MCLR` module.

We are going to optimize the three states of thiophene in C_{2v} symmetry. The inputs are:

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&SEWARD␣&END
Title
Thiophene␣molecule
Symmetry
␣X␣XY
Basis␣set
S.ANO-S...4s3p2d.
S1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-2.1793919255
End␣of␣basis
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.3420838459␣␣␣␣␣␣␣␣␣.1014908659
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣1.3629012233␣␣␣␣␣␣␣␣2.4874875281
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣4.3076765963␣␣␣␣␣␣␣␣-.4350463731
H2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.5065969281␣␣␣␣␣␣␣␣4.1778544652
End␣of␣basis
```

```
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
TITLE
␣Thiophene␣molecule
OCCUPIED
11␣1␣7␣3
ITERATIONS
40
END␣OF␣INPUT
␣&RASSCF␣&END
LUMORB
TITLE
␣Thiophene␣molecule␣1␣1A1
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣0␣␣␣␣7␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
ITER
50,25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIREstart
TITLE
␣Thiophene␣molecule␣1␣1A1
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣0␣␣␣␣7␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
ITER
50,25
END␣OF␣INPUT
␣&ALASKA␣&END
End␣of␣Input
␣&SLAPAF␣&END
Iterations
20
Thrs
0.5D-06␣1.0D-03
End␣of␣Input
>>>␣ENDDO␣<<<
```

for the ground state. For the two excited states we will replace the `RASSCF` inputs with

```
␣&RASSCF␣&END
LUMORB
*JOBIPH
*CIRESTART
TITLE
␣Thiophene␣molecule␣2␣1A1
SYMMETRY
```

```
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣0␣␣␣␣7␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
CIROOT
2␣2
1␣2
1␣1
LEVSHFT
1.0
ITER
50,25
RLXRoot
2
END␣OF␣INPUT
```

for the 2^1A_1 state. Notice that we are doing a SA-CASSCF calculation including two roots, therefore we must use the keyword RLXROOT within the `RASSCF` input to specify for which state we want the root. We have also

```
␣&RASSCF␣&END
LUMORB
*JOBIPH
*CIRESTART
TITLE
␣Thiophene␣molecule␣1␣1B2
SYMMETRY
␣␣␣␣2
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣0␣␣␣␣7␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣3
LEVSHFT
1.0
ITER
50,25
END␣OF␣INPUT
```

for the 1^1B_2 state.

To help the program to converge we can include one or more initial `RASSCF` inputs in the input file. The following is an example for the calculation of the of the $3^1A'$ state of thiophene (C_s symmetry) with a previous calculation of the ground state to have better starting orbitals. The option SALA equal to three is used to relax the CASSCF orbitals for the exact root which we are interested.

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&SEWARD␣&END
Title
Thiophene␣molecule
Symmetry
␣X
```

```
Basis␣set
S.ANO-S...4s3p2d.
S1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-2.1174458547
End␣of␣basis
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.4102089951␣␣␣␣␣␣␣␣␣.1119410701
C1b␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-2.4102089951␣␣␣␣␣␣␣␣␣.1119410701
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣1.3751924147␣␣␣␣␣␣␣␣2.7088559532
C2b␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-1.3751924147␣␣␣␣␣␣␣␣2.7088559532
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣4.3643321746␣␣␣␣␣␣␣␣-.4429940876
H1b␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-4.3643321746␣␣␣␣␣␣␣␣-.4429940876
H2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣2.5331491787␣␣␣␣␣␣␣␣4.3818833166
H2b␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣-2.5331491787␣␣␣␣␣␣␣␣4.3818833166
End␣of␣basis
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
TITLE
␣Thiophene␣molecule
OCCUPIED
18␣4
ITERATIONS
40
END␣OF␣INPUT
␣&RASSCF␣&END
LUMORB
TITLE
␣Thiophene␣molecule␣1A’
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣18␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣5
CIROOT
1␣1
1
LEVSHFT
1.0
ITER
50,25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
␣&RASSCF␣&END
JOBIPH
TITLE
␣Thiophene␣molecule␣3␣1A’
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣18␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣5
CIROOT
```

```
3␣3
1␣2␣3
1␣1␣1
LEVSHFT
1.0
ITER
50,25
RLXRoot
3
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
TITLE
␣Thiophene␣molecule␣3␣1A'
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣␣6␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣18␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣5
CIROOT
3␣3
1␣2␣3
1␣1␣1
LEVSHFT
1.0
ITER
50,25
RLXRoot
3
END␣OF␣INPUT
␣&ALASKA␣&END
End␣of␣Input
␣&SLAPAF␣&END
End␣of␣Input
>>>␣ENDDO␣<<<
```

It should be remembered that geometry optimizations for excited states are difficult. Not only can it be difficult to converge the corresponding `RASSCF` calculation, but we must also be sure that the order of the states does not change during the optimization of the geometry. This is not uncommon and the optimization must be followed by the user.

Sometimes may be interesting to follow the path of the optimization by looking at each one of the output files generated by *MOLCAS*Ȧll the iterative information is stored in the input file if the "Set Output File" command as not used. If it was used the output files of each complete iteration are stored in the $WorkDir directory under the names *`1.save.$iter`*, for instance: *`1.save.1`*, *`1.save.2`*, etc. You should not remove the $WorkDir directory if you want to keep them.

5.2.3 Restrictions in symmetry or geometry.

Optimizing with geometrical constraints.

A common situation in geometry optimizations is to have one or several coordinates fixed or constrained and vary the remaining coordinates. As an example we will take the biphenyl molecule, two benzene moieties bridged by a single bond. The ground state of the molecule is not planar. One benzene group is twisted by 44^{o} degrees with respect to the other [30]. We can use this example to perform two types of restricted optimizations. The simplest way to introduce constraints is to give a coordinate a fixed value and let the other coordinates to be optimized. For instance, let's fix the dihedral angle between both benzenes to be fixed to 44^{o} degrees. Within this restriction, the remaining coordinates will be fully optimized. The CONSTRAINTS keyword in the program `SLAPAF` will take care of the restriction. The input could be:

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&SEWARD␣&END
Title
␣Biphenyl␣twisted␣D2
Symmetry
␣XY␣XZ
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣1.4097582886␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣␣.0000000000
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣2.7703009377␣␣␣␣␣␣␣␣␣2.1131321616␣␣␣␣␣␣␣␣␣␣.8552434921
C3␣␣␣␣␣␣␣␣␣␣␣␣␣␣5.4130377085␣␣␣␣␣␣␣␣␣2.1172148045␣␣␣␣␣␣␣␣␣␣.8532344474
C4␣␣␣␣␣␣␣␣␣␣␣␣␣␣6.7468359904␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H2␣␣␣␣␣␣␣␣␣␣␣␣␣␣1.7692261798␣␣␣␣␣␣␣␣␣3.7578798540␣␣␣␣␣␣␣␣␣1.5134152112
H3␣␣␣␣␣␣␣␣␣␣␣␣␣␣6.4188773347␣␣␣␣␣␣␣␣␣3.7589592975␣␣␣␣␣␣␣␣␣1.5142479153
H4␣␣␣␣␣␣␣␣␣␣␣␣␣␣8.7821560635␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
TITLE
␣Biphenyl␣twisted␣D2
OCCUPIED
␣12␣9␣9␣11
ITERATIONS
50
END␣OF␣INPUT
␣&RASSCF␣&END
LUMORB
TITLE
␣Biphenyl␣twisted␣D2
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣12␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣7␣␣␣␣7␣␣␣10
RAS2
␣␣␣␣2␣␣␣␣4␣␣␣␣4␣␣␣␣2
ITER
50,25
```

```
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
TITLE
␣Biphenyl␣twisted␣D2
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣12␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣7␣␣␣␣7␣␣␣10
RAS2
␣␣␣␣2␣␣␣␣4␣␣␣␣4␣␣␣␣2
ITER
50,25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
␣&ALASKA␣&END
End␣of␣input
␣&SLAPAF␣&END
Constraints
d1␣=␣Dihedral␣C2␣C1␣C1(XY)␣C2(XY)
Values
d1␣=␣44.4␣degrees
End␣of␣Constraints
Iterations
30
End␣of␣Input
>>>␣ENDDO␣<<<
```

One important consideration about the constraint. You do not need to start at a geometry having the exact value for the coordinate you have selected (44.4 degrees for the dihedral angle here). The optimization will lead you to the right solution. On the other hand, if you start exactly with the dihedral being 44.4 deg the code does not necessarily will freeze this value in the first iterations, but will converge to it at the end. Therefore, it may happen that the value for the dihedral differs from the selected value in the initial iterations. You can follow the optimization steps in the $WorkDir directory using the MOLDEN files generated automatically by MOLCAS.

Now we will perform the opposite optimization: we want to optimize the dihedral angle relating both benzene units but keep all the other coordinates fixed. We could well use the same procedure as before adding constraints for all the remaining coordinates different from the interesting dihedral angle, but to build the input would be tedious. Therefore, instead of keyword Constraints we will make use of the keywords Vary and Fix.

The input file should be:

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&SEWARD␣&END
Title
␣Biphenyl␣twisted␣D2
Symmetry
␣XY␣XZ
Basis␣set
C.ANO-S...3s2p1d.
```

```
C1␣␣␣␣␣␣␣␣␣␣␣␣␣1.4097582886␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000
C2␣␣␣␣␣␣␣␣␣␣␣␣␣2.7703009377␣␣␣␣␣␣␣␣2.1131321616␣␣␣␣␣␣␣␣␣.8552434921
C3␣␣␣␣␣␣␣␣␣␣␣␣␣5.4130377085␣␣␣␣␣␣␣␣2.1172148045␣␣␣␣␣␣␣␣␣.8532344474
C4␣␣␣␣␣␣␣␣␣␣␣␣␣6.7468359904␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H2␣␣␣␣␣␣␣␣␣␣␣␣␣1.7692261798␣␣␣␣␣␣␣␣3.7578798540␣␣␣␣␣␣␣␣1.5134152112
H3␣␣␣␣␣␣␣␣␣␣␣␣␣6.4188773347␣␣␣␣␣␣␣␣3.7589592975␣␣␣␣␣␣␣␣1.5142479153
H4␣␣␣␣␣␣␣␣␣␣␣␣␣8.7821560635␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
TITLE
␣Biphenyl␣twisted␣D2
OCCUPIED
␣12␣9␣9␣11
ITERATIONS
50
END␣OF␣INPUT
␣&RASSCF␣&END
LUMORB
TITLE
␣Biphenyl␣twisted␣D2
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣12␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣7␣␣␣␣7␣␣␣10
RAS2
␣␣␣␣2␣␣␣␣4␣␣␣␣4␣␣␣␣2
ITER
50,25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
TITLE
␣Biphenyl␣twisted␣D2
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣12␣␣␣␣0␣␣␣␣0
INACTIVE
␣␣␣11␣␣␣␣7␣␣␣␣7␣␣␣10
RAS2
␣␣␣␣2␣␣␣␣4␣␣␣␣4␣␣␣␣2
ITER
50,25
END␣OF␣INPUT
!cp␣$Project.JobIph␣$Project.JobOld
␣&ALASKA␣&END
End␣of␣input
␣&SLAPAF␣&END
Internal␣coordinates
b1␣=␣Bond␣C1␣C1(XY)
b2␣=␣Bond␣C1␣C2
b3␣=␣Bond␣C2␣C3
```

```
b4␣=␣Bond␣C3␣C4
h1␣=␣Bond␣C2␣H2
h2␣=␣Bond␣C3␣H3
h3␣=␣Bond␣C4␣H4
a1␣=␣Angle␣C2␣C1␣C1(XY)
a2␣=␣Angle␣C1␣C2␣C3
a3␣=␣Angle␣C1␣C2␣H2
a4␣=␣Angle␣C2␣C3␣H3
phi␣=␣Dihedral␣C2␣C1␣C1(XY)␣C2(XY)
d1␣=␣Dihedral␣H2␣C2␣C1␣C1(XY)
d2␣=␣OutOfP␣C3␣C1(XY)␣C1␣C2
d3␣=␣Dihedral␣H3␣C3␣C2␣H2
Vary
␣phi␣=␣1.0␣phi
Fix
␣b1␣=␣1.0␣b1
␣b2␣=␣1.0␣b2
␣b3␣=␣1.0␣b3
␣b4␣=␣1.0␣b4
␣h1␣=␣1.0␣h1
␣h2␣=␣1.0␣h2
␣h3␣=␣1.0␣h3
␣a1␣=␣1.0␣a1
␣a2␣=␣1.0␣a2
␣a3␣=␣1.0␣a3
␣a4␣=␣1.0␣a4
␣d1␣=␣1.0␣d1
␣d2␣=␣1.0␣d2
␣d3␣=␣1.0␣d3
End␣of␣Internal
Iterations
30
End␣of␣Input
>>>␣ENDDO␣<<<
```

To be able to optimize the molecule in that way a D_2 symmetry has to be used. In the definition of the internal coordinates we can use an out-of-plane coordinate: C2 C2(xy) C1(xy) C1 or a dihedral angle C2 C1 C1(xy) C2(xy). In this case there is no major problem but in general one has to avoid as much as possible to define dihedral angles close to $180\,^o$ (trans conformation). The `SLAPAF` program will warn about this problem if necessary. In the present example, angle 'phi' is the angle to vary while the remaining coordinates are frozen. All this is only a problem in the user-defined internal approach, not in the non-redundant internal approach used by default in the program. In case we do not have the coordinates from a previous calculation we can always run a simple calculation with one iteration in the `SLAPAF` program.

It is not unusual to have problems in the relaxation step when one defines internal coordinates. Once the program has found that the definition is consistent with the molecule and the symmetry, it can happen that the selected coordinates are not the best choice to carry out the optimization, that the variation of some of the coordinates is too large or maybe some of the angles are close to their limiting values ($\pm 180\,^o$ for Dihedral angles and $\pm 90\,^o$ for Out of Plane angles). The SLAPAF program will inform about these problems. Most of the situations are solved by re-defining the coordinates, changing the basis set or the geometry if possible, or even freezing some of the coordinates. One easy solution is to froze this particular coordinate and optimize, at least partially, the other as an initial step to a full optimization. It can be recommended to change the definition of the coordinates from internal to Cartesian.

Optimizing with symmetry restrictions.

Presently, *MOLCAS* is prepared to work in the point groups C_1, C_i, C_s, C_2, D_2, C_{2h}, C_{2v}, and D_{2h}. To have the wave functions or geometries in other symmetries we have to restrict orbital rotations or geometry relaxations specifically. We have shown how to in the RASSCF program by using the SUPSYM option. In a geometry optimization we may also want to restrict the geometry of the molecule to other symmetries. For instance, to optimize the benzene molecule which belongs to the D_{6h} point group we have to generate the integrals and wave function in D_{2h} symmetry, the highest group available, and then make the appropriate combinations of the coordinates chosen for the relaxation in the SLAPAF program, as is shown in the manual.

As an example we will take the ammonia molecule, NH_3. There is a planar transition state along the isomerization barrier between two pyramidal structures. We want to optimize the planar structure restricted to the D_{3h} point group. However, the electronic wave function will be computed in C_s symmetry (C_{2v} is also possible) and will not be restricted, although it is possible to do that in the `RASSCF` program.

The input for such a geometry optimization is:

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&SEWARD␣&END
Title
␣NH3,␣planar
Symmetry
␣Z
Basis␣Set
N.ANO-L...4s3p2d.
N␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣Basis
Basis␣set
H.ANO-L...3s2p.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣1.9520879910␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000
H2␣␣␣␣␣␣␣␣␣␣␣␣␣-.9760439955␣␣␣␣␣␣␣␣1.6905577906␣␣␣␣␣␣␣␣␣.0000000000
H3␣␣␣␣␣␣␣␣␣␣␣␣␣-.9760439955␣␣␣␣␣␣␣-1.6905577906␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣Basis
End␣of␣Input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
Title
␣NH3,␣planar
Occupied
␣4␣1
Iterations
40
End␣of␣Input
␣&RASSCF␣&END
LUMORB
Title
␣NH3,␣planar
Symmetry
1
Spin
1
Nactel
␣8␣␣0␣␣0
INACTIVE␣ORBITALS
␣1␣0
RAS2␣ORBITALS
␣6␣2
```

```
ITER
50,20
End␣of␣Input
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
Title
␣NH3,␣planar
Symmetry
1
Spin
1
Nactel
␣8␣␣0␣␣0
INACTIVE␣ORBITALS
␣1␣0
RAS2␣ORBITALS
␣6␣2
ITER
50,20
End␣of␣Input
!cp␣$Project.JobIph␣$Project.JobOld
␣&ALASKA␣&END
End␣of␣input
␣&SLAPAF␣&END
Internal␣coordinates
b1␣=␣Bond␣N␣H1
b2␣=␣Bond␣N␣H2
b3␣=␣Bond␣N␣H3
a1␣=␣Angle␣H1␣N␣H2
a2␣=␣Angle␣H1␣N␣H3
Vary
r1␣=␣1.0␣b1␣+␣1.0␣b2␣+␣1.0␣b3
Fix
r2␣=␣1.0␣b1␣-␣1.0␣b2
r3␣=␣1.0␣b1␣-␣1.0␣b3
a1␣=␣1.0␣a1
a2␣=␣1.0␣a2
End␣of␣internal
Iterations
20
End␣of␣input
>>>␣ENDDO␣<<<
```

All four atoms are in the same plane. Working in C_s, planar ammonia has five degrees of freedom. Therefore we must define five independent internal coordinates, in this case the three N-H bonds and two of the three angles H-N-H. The other is already defined knowing the two other angles. Now we must define the varying coordinates. The bond lengths will be optimized, but all three N-H distances must be equal. First we define (see definition in the previous input) coordinate $r1$ equal to the sum of all three bonds; then, we define coordinates $r2$ and $r3$ and keep them fixed. $r2$ will ensure that $bond1$ is equal to $bond2$ and $r3$ will assure that $bond3$ is equal to $bond1$. $r2$ and $r3$ will have a zero value. In this way all three bonds will have the same length. As we want the system constrained into the D_{3h} point group, the three angles must be equal with a value of 120 degrees. This is their initial value, therefore we simply keep coordinates $ang1$ and $ang2$ fixed. The result is a D_{3h} structure:

```
  ********************************************
  * Values of primitive internal coordinates *
  ********************************************
```

```
  B1    : Bond Length=    1.8957/ bohr
  B2    : Bond Length=    1.8957/ bohr
  B3    : Bond Length=    1.8957/ bohr
  A1     : Angle=  120.0000/degree,     2.0944/rad
  A2     : Angle=  120.0000/degree,     2.0944/rad
```

In a simple case like this an optimization without restrictions would also end up in the same symmetry as the initial input.

5.2.4 Optimizing with Z-Matrix.

An alternative way to optimize a structure with geometrical and/or symmetrical constraints is to combine the Z-Matrix definition of the molecular structure used for the program `SEWARD` with a coherent definition for the INTERNAL COORDINATED used in the optimization by program `SLAPAF`.

Here is an examples of optimization of the methyl carbanion. Note that the wavefunction is calculated within the C_s symmetry but the geometry is optimized within the C_{3v} symmetry throught the ZMAT and the INTERNAL COORDINATES definitions.

```
>>>  Set  MaxIter  500  <<<
>>>  Do  While  <<<
 &SEWARD &END
Symmetry
 Y
ZMAT
H.Aug-cc-pVDZ....
C.Aug-cc-pVDZ....
C1
X2   1  1.00
H3   1  1.09   2 105.
H4   1  1.09   2 105.    3  120.
End of input
 &SCF &END
Charge
 -1
End of input
 &ALASKA &END
End of input
 &SLAPAF &END
Internal Coordinates
CX2  = Bond C1 X2
CH3  = Bond C1 H3
CH4  = Bond C1 H4
XCH3 = Angle X2 C1 H3
XCH4 = Angle X2 C1 H4
DH4  = Dihedral H3 X2 C1 H4
Vary
SumCH34  = 1. CH3  +2. CH4
SumXCH34 = 1. XCH3 +2. XCH4
Fix
rCX2  = 1.0 CX2
DifCH34  = 2. CH3  -1. CH4
DifXCH34 = 2. XCH3 -1. XCH4
dDH4  = 1.0 DH4
End of Internal
PRFC
Iterations
 10
End of input
>>>  EndDo  <<<
```

Note that the *dummy* atom X2 is used to define the Z axis and the planar angles for the hydrogen atoms. The linear combinations of bond distances and planar angles in the expression in the VARY and FIX sections are used to impose the C_{3v} symmetry.

Another examples where the wavefunction and the geometry can be calculated within different symmetry groups is benzene. In this case, the former uses D_{2h} symmetry and the latter D_{6h} symmetry. Two special atoms are used: the *dummy* X1 atom defines the center of the molecule while the *ghost* Z2 atom is used to define the C_6 rotational axis (and the Z axis).

```
>>>  Set  MaxIter  500  <<<
>>>  Do  While  <<<
 &SEWARD &END
Symmetry
 X Y Z
ZMAT
H.ANO-S...2s.
C.ANO-S...3s2p.
X1
Z2   1  1.00
C3   1  1.3915   2  90.
C4   1  1.3915   2  90.    3  60.
H5   1  2.4715   2  90.    3   0.
H6   1  2.4715   2  90.    3  60.
End of input
 &SCF &END
End of input
 &ALASKA &END
End of input
 &SLAPAF &END
Internal Coordinates
XC3 = Bond X1 C3
XC4 = Bond X1 C4
XH5 = Bond X1 H5
XH6 = Bond X1 H6
CXC = Angle C3 X1 C4
HXH = Angle H5 X1 H6
Vary
SumC = 1.0 XC3 + 2.0 XC4
SumH = 1.0 XH5 + 2.0 XH6
Fix
DifC = 2.0 XC3 - 1.0 XC4
DifH = 2.0 XH5 - 1.0 XH6
aCXC = 1.0 CXC
aHXH = 1.0 HXH
End of Internal
PRFC
Iterations
 10
End of input
>>>  EndDo  <<<
```

Note that the *ghost* atom Z2 is used to define the geometry within the Z-Matrix but it does not appear in the INTERNAL COORDINATES section. On the other hand, the *dummy* atom X1 rapresents the center of the molecule and it is used in the INTERNAL COORDINATES section.

5.2.5 CASPT2 optimizations

For systems showing a clear multiconfigurational nature, the CASSCF treatment on top of the HF results is of crucial importance in order to recover the large non dynamical correlation effects. On the other hand, ground-state geometry optimizations of closed shell systems are not exempt from non dynamical correlation effects. In general, molecules containing π-electrons suffer from significant effects of non dynamical correlation, even more in presence of conjugated groups. Several studies on systems with delocalized bonds have shown the effectiveness of the CASSCF approach in reproducing the main geometrical parameters with high accuracy [31, 32, 33].

However, pronounced effects of dynamical correlation often occur in systems with π-electrons, especially in combination with polarized bonds. An example is given by the C=O bond length, which is known to be very sensitive to an accurate description of the dynamical correlation effects [34]. We will show now that the inherent limitations of the CASSCF method can be successfully overcome by employing a CASPT2 geometry optimization, which uses a numerical gradient procedure of recent implementation. A suitable molecule for this investigation is acrolein. As many other conjugated aldehydes and ketones, offers an example of *s-cis/s-trans* isomerism (Figure 5.4). Due to the resonance between various structures involving π electrons, the bond order for the C-C bond is higher than the one for a non-conjugated C-C single bond. This partial double-bond character restricts the rotation about such a bond, giving rise to the possibility of geometrical isomerism, analogue to the *cis–trans* one observed for conventional double bonds.

A `CASPT2` geometry optimization in *MOLCAS* can be performed using `AUTO` program. A possible input for the CASPT2 geometry optimization of the *s-trans* isomer is displayed below.

```
*
*␣␣␣␣Start␣Structure␣calculation
*
>>>␣SET␣MAXITER␣500␣<<<
>>>␣SET␣OUT␣FILE␣<<<
>>>>>>>>>>>>>␣Do␣while␣<<<<<<<<<<<<
&SEWARD␣&END
Title
␣Acrolein␣Cs␣symmetry␣-␣transoid
Symmetry
␣x
Basis␣set
O.ANO-L...4s3p1d.
O␣␣␣␣␣␣␣0.0000000000␣␣-1.319834␣␣-1.216012␣/Angstrom
end␣of␣basis
Basis␣set
C.ANO-L...4s3p1d.
C1␣␣␣␣␣␣0.0000000000␣␣-0.743740␣␣-0.151008␣/Angstrom
C2␣␣␣␣␣␣0.0000000000␣␣␣0.719660␣␣␣0.000000␣/Angstrom
C3␣␣␣␣␣␣0.0000000000␣␣␣1.280044␣␣␣1.212021␣/Angstrom
end␣of␣basis
Basis␣set
H.ANO-L...2s1p.
H1␣␣␣␣␣0.0000000000␣␣␣-1.307048␣␣␣0.809732␣/Angstrom
H2␣␣␣␣␣0.0000000000␣␣␣␣1.306638␣␣-0.915400␣/Angstrom
H3␣␣␣␣␣0.0000000000␣␣␣␣0.666909␣␣␣2.111907␣/Angstrom
H4␣␣␣␣␣0.0000000000␣␣␣␣2.356392␣␣␣1.355776␣/Angstrom
end␣of␣basis
END␣OF␣INPUT
>>>>>>>>␣IF␣(␣ITER␣=␣1␣)␣<<<<<<<<<<<
␣&SCF␣&END
Title
Acrolein␣Cs␣symmetry
Occupied
```

```
*The␣symmetry␣species␣are␣a’␣␣a’’
␣13␣2
End␣of␣input
␣&RASSCF␣&END
LUMORB
Title
Acrolein␣ground␣state
Spin
␣1
Symmetry
␣1
nActEl
␣4␣0␣0
Inactive
*The␣symmetry␣species␣are␣a’␣␣a’’
␣13␣0
Ras2
␣0␣4
CIroot
␣1␣1
␣1
THRS
1.0e-06␣1.0e-04␣1.0e-04
ITERation
␣100␣100
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
>>>>>>>␣ENDIF␣<<<<<<<<<<<<<<<<<<<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
Title
Acrolein␣ground␣state
Spin
␣1
Symmetry
␣1
nActEl
␣4␣0␣0
Inactive
*The␣symmetry␣species␣are␣a’␣␣a’’
␣13␣0
Ras2
␣0␣4
CIroot
␣1␣1
␣1
THRS
1.0e-06␣1.0e-04␣1.0e-04
ITERation
␣100␣100
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
␣&CASPT2␣&END
Title
␣acrolein␣s-trans
Maxit
␣20
Lroot
␣1
End␣of␣input
␣&Alaska␣&END
End␣of␣input
␣&SLAPAF␣&END
Iterations
␣20
```

```
Thrs
␣1.0d-8␣1.0d-4
End␣of␣input
>>>>>>>>>>>>>␣ENDDO␣␣<<<<<<<<<<<<<<
```

Experimental investigations assign a planar structure for both the isomers. We can take advantage of this result and use a C_s symmetry throughout the optimization procedure. Moreover, the choice of the active space is suggested by previous calculations on analogous systems. The active space contains 4 π MOs /4 π electrons, thus what we will call shortly a π-CASPT2 optimization.

The structure of the input follows the trends already explained in other geometry optimizations, that is, loops over the set of programs ending with `SLAPAF`. Notice that CASPT2 optimizations require obviously the `CASPT2` input, but also the input for the `ALASKA` program, which computes the gradient numerically. Apart from that, a CASPT2 optimization input is identical to the corresponding CASSCF input. With the present implementation of the CASPT2 numerical gradients procedure, the output of our calculation will contain by default all the intermediate outputs at each point of the grid. Even for a calculation with a small number of internal coordinates, slowly convergent situations can end up in an undesirably huge output file. To avoid this annoying circumstance, the keyword SET OUTPUT FILE of the `AUTO` script can be used. In this case, the output will be split into several files each corresponding the the whole calculations done in a single iteration of the geometry optimization. The keyword SET OVER can be also useful especially when we want to save disk space. It replaces the output of each iteration with the one of the subsequent, overriding the previous output file. The output file corresponding to the final iteration, contains the summary concerning each iterations. For our example the résumé is the following:

```
************************************************************************************************
***************** Energy Statistics for Geometry Optimization **********************************
************************************************************************************************
                 Energy     Grad     Grad             Step             Estimated Hess Geom Hess
Iter  Energy     Change     Norm     Max      Element  Max      Element Final Ene Index Upd Update
 1 -191.46171057 0.00000000 0.028786 0.015590 nrc001  0.019477 nrc001 -191.46189776 0 RF(S) None
 2 -191.46195237-0.00024180 0.010010 0.005154 nrc001  0.012652 nrc001 -191.46198941 0 RF(S) BFGS
 3 -191.46198838-0.00003601 0.001320-0.000670 nrc012 -0.002159 nrc012 -191.46198994 0 RF(S) BFGS
 4 -191.46199164-0.00000326 0.000661-0.000265 nrc012  0.000527 nrc004 -191.46199187 0 RF(S) BFGS
 5 -191.46199182-0.00000017 0.000435-0.000184 nrc012 -0.000903 nrc012 -191.46199231 0 RF(S) BFGS
 6 -191.46199215-0.00000034 0.000190 0.000094 nrc010  0.000982 nrc010 -191.46199230 0 RF(S) BFGS
 7 -191.46199217-0.00000002 0.000086 0.000028 nrc010  0.000570 nrc012 -191.46199220 0 RF(S) BFGS
 8 -191.46199214 0.00000003 0.000134 0.000061 nrc010  0.000464 nrc010 -191.46199217 0 RF(S) BFGS
 9 -191.46199216-0.00000002 0.000079 0.000028 nrc010 -0.000232 nrc010 -191.46199215 0 RF(S) BFGS
         Cartesian Displacements            Gradient in internals
       Value      Threshold  Converged?   Value      Threshold  Converged?
    +----------------------------------+----------------------------------+
RMS + 0.2558E-03  0.4000E-03     Yes   + 0.2291E-04  0.1000E-03     Yes   +
    +----------------------------------+----------------------------------+
Max + 0.3879E-03  0.6000E-03     Yes   + 0.2761E-04  0.1500E-03     Yes   +
    +----------------------------------+----------------------------------+
 Geometry is converged
************************************************************************************************
************************************************************************************************
```

The calculation converges in 9 iterations. At this point it is worth noticing how the convergence of CASPT2 energy is not chosen among the criteria for the convergence of the structure. The final structure is in fact decided by checking the Cartesian displacements and the gradient in non-redundant internal coordinates.

CASPT2 optimizations are expensive. Notice that they are based on numerical gradients and many point-wise calculations are needed. In particular, double-sided gradients are computed in Cartesian. Therefore, each macro-iteration in the optimization requires 2*N + 1 Seward/RASSCF/CASPT2 calculations, with N being the Cartesian degrees of freedom. In the present example, acrolein has eight atoms. From each atom, only two Cartesian coordinates are free to move (we are working within the C_s symmetry and the third coordinate is frozen), therefore the total number of Seward/RASSCF/CASPT2 iterations within each macro-iteration is 2*(8*2) + 1, that is, 33. It is not an easy task.

The Table 5.8 displays the equilibrium geometrical parameters computed at the π-CASSCF and π-CASPT2 level of theory for the ground state of both isomers of acrolein. For sake of comparison, Table 5.8 includes experimental data obtained from microwave spectroscopy studies[35]. The computed parameters at π-CASPT2 level are in remarkable agreement with the experimental data. The predicted value of the C=C bond length is very close to the double bond length observed in ethylene. The other C-C bond has a length within the range expected for a C-C single bond: it appears shorter in the s-trans isomer as a consequence of the reduction of steric hindrance between the ethylenic and aldehydic moieties. CASSCF estimates a carbon-oxygen bond length shorter than the experimental value. For π-CASSCF optimization in conjugated systems this can be assumed as a general behavior [36, 34]. To explain such a discrepancy, one may invoke the fact that the C=O bond distance is particularly sensitive to electron correlation effects. The π electron correlation effects included at the π-CASSCF level tend to overestimate bond lengths. However, the lack of σ electron correlation, goes in the opposite direction, allowing shorter bond distances for double bonds. For the C-C double bonds, these contrasting behaviors compensate each other [33] resulting in quite an accurate value for the bond length at the π-CASSCF level. On the contrary, the extreme sensitivity of the C=O bond length to the electron correlation effects, leads to a general underestimation of the C-O double bond lengths, especially when such a bond is part of a conjugated system. It is indeed the effectiveness of the CASPT2 method in recovering dynamical correlation which leads to a substantial improvement in predicting the C-O double bond length.

Table 5.8: Geometrical parameters for the ground state of acrolein

Parameters[a]	π-CASSCF [04/4]		π-CASPT2		Expt.[b]
	s-cis	*s-trans*	*s-cis*	*s-trans*	
C_1=O	1.204	1.204	1.222	1.222	1.219
C_1–C_2	1.483	1.474	1.478	1.467	1.470
C_2=C_3	1.340	1.340	1.344	1.344	1.345
$\angle$ $C_1C_2C_3$	123.0	121.7	121.9	120.5	119.8
$\angle$ C_2C_1O	124.4	123.5	124.5	124.2	-

[a] Bond distances in Å and angles in degrees.

[b] Microwave spectroscopy data from ref. [35]. No difference between s-cis and s-trans isomers is reported

The use of numerical CASPT2 gradients can be extended to all the optimizations available in `SLAPAF`, for instance transition state searches. Use the following input for the water molecule to locate the linear transition state:

```
>>>␣SET␣MAXITER␣500␣<<<
>>>␣SET␣OUT␣FILE␣<<<
␣&SEWARD␣&END
```

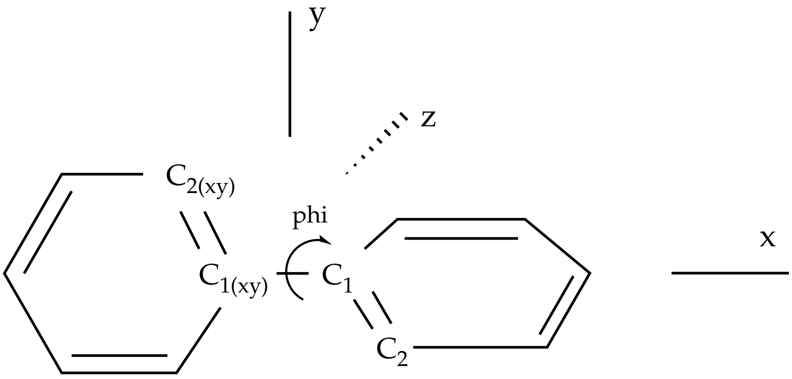


Figure 5.3: Twisted biphenyl molecule

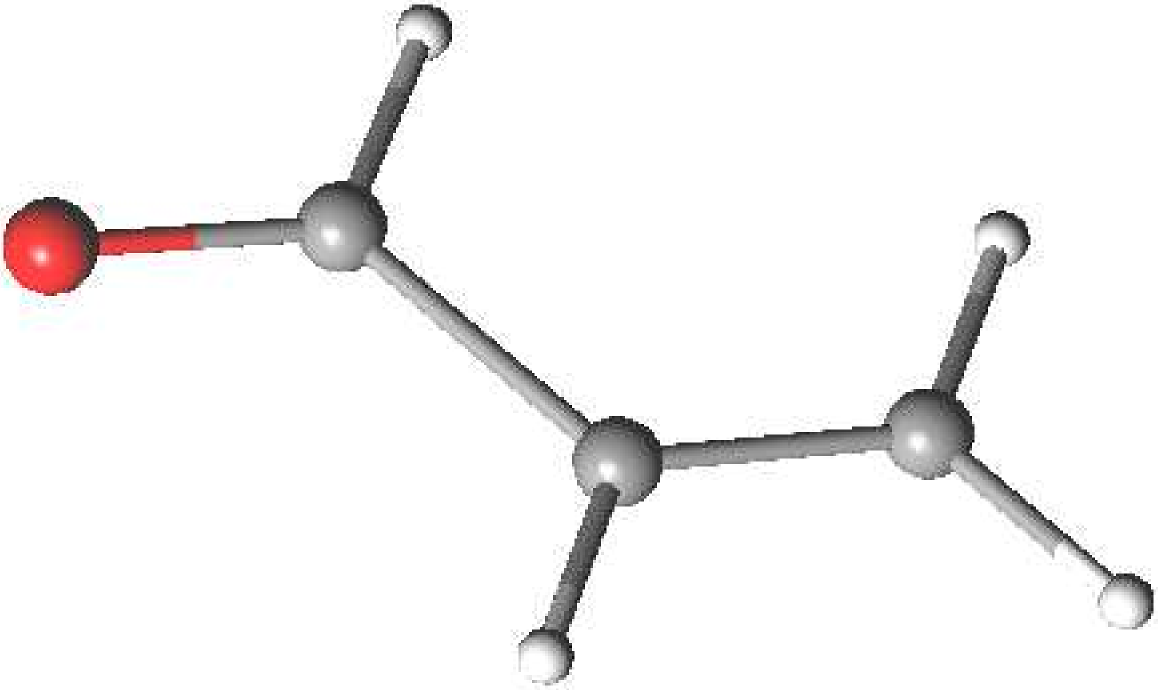


Figure 5.4: Acrolein geometrical isomers

```
Title
␣Water,␣STO-3G␣Basis␣set
Basis␣set
H.STO-3G....
␣␣␣␣␣H1␣␣␣␣␣␣␣␣␣␣␣␣-1.4392565728␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-1.1234014131
␣␣␣␣␣H2␣␣␣␣␣␣␣␣␣␣␣␣␣1.4392565728␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-1.1234014131
End␣of␣basis
Basis␣set
O.STO-3G....
␣␣␣␣␣O␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.1415689960
End␣of␣basis
End␣of␣input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&END
Title
water,␣STO-3g␣Basis␣set
Occupied
5
End␣of␣input
␣&RASSCF␣&END
LumOrb
Nactel
2␣0␣0
Inactive
4
Ras2
1
End␣of␣Input
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣ENDIF␣<<<
␣&RASSCF␣&END
JOBIPH
CIRESTART
Nactel
2␣0␣0
Inactive
4
Ras2
1
End␣of␣Input
!cp␣$Project.JobIph␣$Project.JobOld
␣&CASPT2␣&END
Frozen
1
End␣of␣input
␣&ALASKA␣&END
End␣of␣Input
␣&SLAPAF␣&END
TS
End␣of␣Input
>>>␣ENDDO␣<<<
```

After ten macro-iterations the linear water is reached:

```
*********************************************************************************************
*************** Energy Statistics for Geometry Optimization *********************************
*********************************************************************************************
                Energy     Grad     Grad              Step              Estimated Hess Geom Hess
Iter  Energy    Change     Norm     Max    Element    Max    Element Final Ener Index Upd Upd
  1 -75.00603925 0.00000000 0.000505-0.000333 nrc001  0.149153 nrc003 -75.00437273 1 MFRFS None
  2 -75.00256314 0.00347612 0.033114-0.027092 nrc003  0.145257 nrc003 -75.00034225 1 MFRFS MSP
  3 -74.99310559 0.00945755 0.083775-0.078714 nrc003 -0.184679 nrc002 -74.98597057 1 MFRFS MSP
  4 -74.97219951 0.02090608 0.163015 0.086748 nrc002  0.226701 nrc003 -74.94503565 1 MFRFS MSP
  5 -74.93277784 0.03942168 0.201340 0.123100 nrc002  0.223518 nrc003 -74.88089265 1 MFRFS MSP
  6 -74.89601350 0.03676433 0.148875-0.100716 nrc003  0.230650 nrc003 -74.86658241 1 MFRFS MSP
  7 -74.87796405 0.01804946 0.044034 0.037502 nrc002  0.055719 nrc002 -74.87855019 1 MFRFS MSP
```

```
  8 -74.87878116-0.00081712 0.009948-0.007364 nrc003  0.016283 nrc003 -74.87873039 1 MFRFS MSP
  9 -74.87872319 0.00005797 0.000848-0.000641 nrc001  0.001232 nrc003 -74.87872320 1 MFRFS MSP
 10 -74.87872373-0.00000054 0.000053-0.000048 nrc001 -0.000066 nrc001 -74.87872373 1 MFRFS MSP
         Cartesian Displacements           Gradient in internals
       Value      Threshold  Converged?  Value      Threshold  Converged?
    +---------------------------------+---------------------------------+
 RMS + 0.4174E-04  0.1200E-02     Yes   + 0.3719E-04  0.3000E-03     Yes   +
    +---------------------------------+---------------------------------+
 Max + 0.4665E-04  0.1800E-02     Yes   + 0.4836E-04  0.4500E-03     Yes   +
    +---------------------------------+---------------------------------+
  Geometry is converged
 ********************************************************************************************
 ********************************************************************************************
```

5.3 Computing a reaction path.

Chemists are familiarized with the description of a chemical reaction as a continuous motion on certain path of the potential energy hypersurfaces connecting reactants with products. Those are considered minima in the hypersurface while an intermediate state known as the transition state would be a saddle point of higher energy. The height of the energy barrier separating reactants from products relates to the overall rate of reaction, the positions of the minima along the reaction coordinate give the equilibrium geometries of the species, and the relative energies relate to the thermodynamics of the process. All this is known as transition state theory.

The process to study a chemical reaction starts by obtaining proper geometries for reactants and products, follows by finding the position of the transition state, and finishes by computing as accurately as possible the relative energies relating the position of the species. To perform geometry optimizations searching for true minima in the potential energy surfaces (PES) is by now a well-established procedure (see section 5.2). An stationary point in the PES is characterized by having all the first derivatives of the energy with respect to each one of the independent coordinates equal to zero and the second derivatives larger than zero. First-order saddle points, on the contrary, have their second derivatives lower than zero for one coordinate, that is, they are maxima along this coordinate. A transition state is defined as a saddle point having only one negative second derivative along the specific coordinate known as the reaction coordinate. To simplify the treatment a special set of coordinates known as normal coordinates is defined in a way that the matrix of second derivatives is diagonal. A transition state will have one negative value in the diagonal of such a matrix.

Finally once the reactant, product and transition state geometries have been established one could perform a Intrinsic Reaction Coordinate (IRC) analysis. This to find the energy profile of the reaction and also to establish that the found transition state is connected to the reactant and the product.

5.3.1 Studying a reaction

The localization of the transition state of a reaction is of importance in both a qualitative and quantitative description of the reaction mechanism and the thermodynamics of a reaction. In the following example we will locate the transition state of the proton transfer reaction between the two species in Figs. 5.5 and 5.6. The example selected here is chosen to demonstrate the steps needed to find a transition state. For that sake we have limited our model to the SCF level of theory.

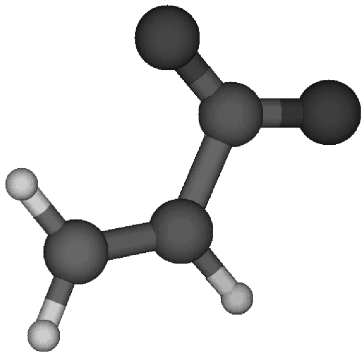


Figure 5.5: Reactant

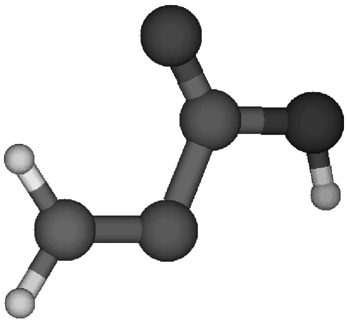


Figure 5.6: Product

Reactant and product

The first step is to establish the two species in equilibrium. These calculations would constitute standard geometry optimizations with the input for the reactant

```
>>>␣Set␣MaxIter␣50␣<<<
>>>␣Do␣while␣<<<
␣&Seward␣&End
Basis␣set
C.cc-pVDZ....
C1␣␣␣␣␣␣␣␣␣␣␣␣-1.9385577715␣␣␣␣␣␣␣␣0.0976565175␣␣␣␣␣␣␣␣0.4007212526
C2␣␣␣␣␣␣␣␣␣␣␣␣-2.4151209200␣␣␣␣␣␣␣-0.0592579424␣␣␣␣␣␣␣␣2.8519334864
C3␣␣␣␣␣␣␣␣␣␣␣␣␣0.7343463765␣␣␣␣␣␣␣␣0.0088689871␣␣␣␣␣␣␣-0.7477660837
End␣of␣Basis
Basis␣set
H.cc-pVDZ....
H1␣␣␣␣␣␣␣␣␣␣␣␣-4.3244501026␣␣␣␣␣␣␣␣0.0091320829␣␣␣␣␣␣␣␣3.6086029352
H2␣␣␣␣␣␣␣␣␣␣␣␣-0.8591520071␣␣␣␣␣␣␣-0.2642180524␣␣␣␣␣␣␣␣4.1663142585
H3␣␣␣␣␣␣␣␣␣␣␣␣-3.4743702487␣␣␣␣␣␣␣␣0.3026128386␣␣␣␣␣␣␣-0.9501874771
End␣of␣Basis
Basis␣set
O.cc-pVDZ....
O1␣␣␣␣␣␣␣␣␣␣␣␣␣0.7692102769␣␣␣␣␣␣␣␣0.1847569555␣␣␣␣␣␣␣-3.0700425345
O2␣␣␣␣␣␣␣␣␣␣␣␣␣2.4916838932␣␣␣␣␣␣␣-0.2232135341␣␣␣␣␣␣␣␣0.7607580753
End␣of␣Basis
End␣of␣input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&End
Core
Charge
-1.0
End␣of␣Input
>>>␣ENDIF␣<<<
␣&SCF␣&End
LUMORB
Charge
-1.0
End␣of␣Input
␣&Alaska␣&End
End␣of␣Input
␣&Slapaf␣&End
Iterations
20
End␣of␣Input
>>>␣ENDDO␣<<<
```

resulting in the following convergence pattern

```
                      Energy     Grad      Grad          Step           Estimated  Hessian Geom Hessian
Iter     Energy       Change     Norm       Max  Element  Max  Element  Final Energy Index Update Update
 1   -265.09033194 0.00000000 0.091418 0.044965 nrc003 0.069275  nrc003 -265.09529138  0  RF(S)   None
 2   -265.09646330-0.00613136 0.020358 0.008890 nrc003 0.040393  nrc008 -265.09684474  0  RF(S)   BFGS
 3   -265.09693242-0.00046912 0.011611-0.005191 nrc001 0.079285  nrc016 -265.09709856  0  RF(S)   BFGS
 4   -265.09655626 0.00037616 0.020775-0.010792 nrc016-0.070551  nrc016 -265.09706324  0  RF(S)   BFGS
 5   -265.09706308-0.00050682 0.003309-0.001628 nrc003-0.010263  nrc017 -265.09707265  0  RF(S)   BFGS
 6   -265.09707056-0.00000747 0.000958-0.000450 nrc011 0.017307  nrc017 -265.09707924  0  RF(S)   BFGS
 7   -265.09706612 0.00000444 0.002451 0.001148 nrc003-0.011228  nrc018 -265.09706837  0  RF(S)   BFGS
 8   -265.09707550-0.00000938 0.000516 0.000220 nrc001-0.004017  nrc014 -265.09707591  0  RF(S)   BFGS
 9   -265.09707586-0.00000036 0.000286 0.000104 nrc001 0.002132  nrc017 -265.09707604  0  RF(S)   BFGS
```

and for the product the input

```
>>>␣Set␣MaxIter␣50␣<<<
```

```
>>>␣Do␣while␣<<<
␣&Seward␣&End
Basis␣set
C.cc-pVDZ....
C1␣␣␣␣␣␣␣␣␣␣␣␣-2.0983667072␣␣␣␣␣␣␣␣0.1000525724␣␣␣␣␣␣␣␣0.5196668948
C2␣␣␣␣␣␣␣␣␣␣␣␣-2.1177298783␣␣␣␣␣␣␣-0.0920244467␣␣␣␣␣␣␣␣3.0450747772
C3␣␣␣␣␣␣␣␣␣␣␣␣␣0.5639781563␣␣␣␣␣␣␣␣0.0024463770␣␣␣␣␣␣␣-0.5245225314
End␣of␣Basis
Basis␣set
H.cc-pVDZ....
H1␣␣␣␣␣␣␣␣␣␣␣␣-3.8870548756␣␣␣␣␣␣␣-0.0558560582␣␣␣␣␣␣␣␣4.1138131865
H2␣␣␣␣␣␣␣␣␣␣␣␣-0.4133953535␣␣␣␣␣␣␣-0.2946498869␣␣␣␣␣␣␣␣4.2050068095
H3␣␣␣␣␣␣␣␣␣␣␣␣-1.3495534119␣␣␣␣␣␣␣␣0.3499572533␣␣␣␣␣␣␣-3.3741881412
End␣of␣Basis
Basis␣set
O.cc-pVDZ....
O1␣␣␣␣␣␣␣␣␣␣␣␣␣0.5100106099␣␣␣␣␣␣␣␣0.2023808294␣␣␣␣␣␣␣-3.0720173949
O2␣␣␣␣␣␣␣␣␣␣␣␣␣2.5859515474␣␣␣␣␣␣␣-0.2102046338␣␣␣␣␣␣␣␣0.4795705925
End␣of␣Basis
End␣of␣input
>>>␣IF␣(␣ITER␣=␣1␣)␣<<<
␣&SCF␣&End
Core
Charge
-1.0
End␣of␣Input
>>>␣ENDIF␣<<<
␣&SCF␣&End
LUMORB
Charge
-1.0
End␣of␣Input
␣&Alaska␣&End
End␣of␣Input
␣&Slapaf␣&End
Iterations
20
End␣of␣Input
>>>␣ENDDO␣<<<
```

resulting in the following convergence pattern

```
                      Energy     Grad      Grad          Step           Estimated  Hessian Geom Hessian
Iter     Energy       Change     Norm       Max  Element  Max  Element  Final Energy Index Update Update
 1   -265.02789209 0.00000000 0.062885-0.035740 nrc006-0.060778  nrc006 -265.02939600  0  RF(S)   None
 2   -265.02988181-0.00198972 0.018235-0.011496 nrc006-0.023664  nrc006 -265.03004886  0  RF(S)   BFGS
 3   -265.03005329-0.00017148 0.001631-0.000978 nrc009-0.015100  nrc017 -265.03006082  0  RF(S)   BFGS
 4   -265.03004953 0.00000376 0.002464-0.000896 nrc014 0.013752  nrc017 -265.03006022  0  RF(S)   BFGS
 5   -265.03006818-0.00001865 0.001059 0.000453 nrc013-0.007550  nrc014 -265.03007064  0  RF(S)   BFGS
 6   -265.03006524 0.00000294 0.001800 0.000778 nrc014 0.006710  nrc014 -265.03007032  0  RF(S)   BFGS
 7   -265.03006989-0.00000465 0.000381 0.000190 nrc005 0.003078  nrc016 -265.03007014  0  RF(S)   BFGS
 8   -265.03006997-0.00000008 0.000129-0.000094 nrc016-0.001305  nrc017 -265.03007003  0  RF(S)   BFGS
```

The computed reaction energy is estimated to about 42 kcal/mol at this level of theory.

Transition state optimization

To locate the transition state it is important to identify the reaction coordinate. In our case here we note that the significant reaction coordinates are the bond distances between C1 and H3, and O1 and H3. In the location of the transition state we will start from the geometry of the reactant for which the O1-H3 bond distance is 2.51 Ångström. We will conduct the search in a number of constrained geometry optimizations in which we step by step reduce the O1-H3 distance towards the distance in the product of 0.95 Ångström. The selected series is 2.0, 1.5, 1.3, and 1.0 Ångström. To constraint the O1-H3 bond distance we modify the input to the relaxation module, SLAPAF, as follows

```
␣&Slapaf␣&End
Iterations
20
Constraint
R1␣=␣Bond␣H3␣O1
Value
R1␣=␣2.0␣Angstrom
End␣of␣Constraint
FindTS
End␣of␣Input
```

This will correspond to the input for SLAPAF for the first of the series of constraint geometry optimization. However, note the keyword FindTS. This keyword will make the SLAPAF module switch from a constrained geometry optimization to a transition state geometry optimization if the updated geometrical Hessian contains one negative eigenvalue. It is of course our hope that during the series of constrained geometry optimizations that we will run into this situation and find the transition state. The convergence pattern for the first constrained optimization is

```
                      Energy     Grad     Grad          Step           Estimated  Hessian Geom Hessian
Iter     Energy       Change     Norm      Max  Element  Max  Element  Final Energy Index Update Update
 1   -265.09707600 0.00000000 0.965614 0.965614 Cns001 0.230366* nrc009 -265.07671229  0  MFRFS  None
 2   -265.08759913 0.00947687 0.216939 0.214768 Cns001 0.081441  nrc012 -265.08946379  0  MFRFS   MSP
 3   -265.08218288 0.00541624 0.014770 0.007032 nrc010 0.019690  nrc010 -265.08242668  0  MFRFS   MSP
 4   -265.08251826-0.00033537 0.003644-0.001560 nrc003 0.005075  nrc002 -265.08254163  0  MFRFS   MSP
 5   -265.08254834-0.00003008 0.001274-0.000907 nrc012 0.026237! nrc016 -265.08257455  0  MFRFS   MSP
 6   -265.08251413 0.00003421 0.003036-0.002420 nrc016-0.024325  nrc016 -265.08254699  0  MFRFS   MSP
 7   -265.08254682-0.00003269 0.000837-0.000426 nrc012 0.012351  nrc017 -265.08255083  0  MFRFS   MSP
 8   -265.08255298-0.00000616 0.000470 0.000238 nrc016-0.005376  nrc017 -265.08255421  0  MFRFS   MSP
 9   -265.08255337-0.00000038 0.000329-0.000154 nrc012-0.004581  nrc014 -265.08255409  0  MFRFS   MSP
10   -265.08255418-0.00000081 0.000206-0.000148 nrc012-0.000886  nrc014 -265.08255425  0  MFRFS   MSP
11   -265.08255430-0.00000013 0.000123-0.000097 nrc012-0.001131  nrc014 -265.08255436  0  MFRFS   MSP
```

Here we note that the Hessian index is zero, i.e. the optimization is a constrained geometry optimization. The final structure is used as the starting geometry for the 2nd constrained optimization at 1.5 Ångström. This optimization did not find a negative eigenvalue either. However, starting the 3rd constrained optimization from the final structure of the 2nd constrained optimization resulted in the convergence pattern

```
                      Energy     Grad     Grad          Step           Estimated  Hessian Geom Hessian
Iter     Energy       Change     Norm      Max  Element  Max  Element  Final Energy Index Update Update
 1   -265.03250948 0.00000000 0.384120 0.377945 Cns001-0.209028* nrc007 -264.99837542 0  MFRFS   None
 2   -265.01103140 0.02147809 0.120709 0.116546 Cns001-0.135181  nrc007 -265.01209656 0  MFRFS   MSP
 3   -265.00341440 0.00761699 0.121043-0.055983 nrc005-0.212301* nrc007 -264.98788416 1  MFRFS   MSP
 4   -264.99451339 0.00890101 0.089986 0.045423 nrc007 0.123178* nrc002 -264.99582814 1  MFRFS   MSP
 5   -264.99707885-0.00256546 0.044095-0.015003 nrc009 0.159069* nrc015 -265.00090995 1  MFRFS   MSP
 6   -264.99892919-0.00185034 0.033489-0.013653 nrc015-0.124146  nrc015 -265.00050567 1  MFRFS   MSP
 7   -265.00031159-0.00138240 0.009416-0.004916 nrc018-0.156924  nrc018 -265.00070286 1  MFRFS   MSP
 8   -265.00019076 0.00012083 0.009057 0.005870 nrc018 0.081240  nrc018 -265.00049408 1  MFRFS   MSP
 9   -265.00049567-0.00030490 0.003380 0.001481 nrc011-0.070124  nrc015 -265.00056966 1  MFRFS   MSP
10   -265.00030276 0.00019291 0.159266-0.159144 Cns001 0.114927! nrc015 -264.99874954 0  MFRFS   MSP
11   -265.00098377-0.00068101 0.031621-0.008700 nrc005-0.101187  nrc007 -265.00046906 1  MFRFS   MSP
12   -265.00050857 0.00047520 0.003360 0.001719 nrc015 0.012580  nrc015 -265.00052069 1  MFRFS   MSP
13   -265.00052089-0.00001233 0.001243-0.000590 nrc017-0.006069  nrc017 -265.00052323 1  MFRFS   MSP
14   -265.00052429-0.00000340 0.000753 0.000259 nrc011-0.002449  nrc018 -265.00052458 1  MFRFS   MSP
```

```
15   -265.00052441-0.00000011 0.000442-0.000136 nrc007 0.003334  nrc018 -265.00052464 1  MFRFS   MSP
16   -265.00052435 0.00000006 0.000397 0.000145 nrc017 0.001628  nrc010 -265.00052459 1  MFRFS   MSP
```

Here a negative Hessian eigenvalue was found at iteration 3. At this point the optimization turn to a normal quasi-Newton Raphson optimization without any constraints. We note that the procedure flips back to a constrained optimization at iteration 10 but is finished as an optimization for a transition state. The predicted activation energy is estimated to 60.6 kcal/mol (excluding vibrational corrections). The computed transition state is depicted in Fig. 5.7.

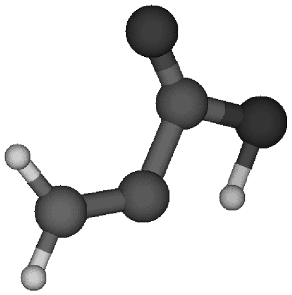


Figure 5.7: Transition state

The remaining issue is if this is a true transition state. This issue can only be resolved by doing a calculation of the analytical Hessian using the MCKINLEY module (execution of the MCLR module is automatic). The corresponding input is

```
␣&Seward␣&End
Basis␣set
C.cc-pVDZ....
C1␣␣␣␣␣␣␣␣␣␣␣␣-1.8937541206␣␣␣␣␣␣␣␣0.0797525492␣␣␣␣␣␣␣␣0.5330826031
C2␣␣␣␣␣␣␣␣␣␣␣␣-2.3239194706␣␣␣␣␣␣␣-0.0748842444␣␣␣␣␣␣␣␣3.0012862573
C3␣␣␣␣␣␣␣␣␣␣␣␣␣0.7556108398␣␣␣␣␣␣␣-0.0065134659␣␣␣␣␣␣␣-0.5801137465
End␣of␣Basis
Basis␣set
H.cc-pVDZ....
H1␣␣␣␣␣␣␣␣␣␣␣␣-4.2196708766␣␣␣␣␣␣␣-0.0106202053␣␣␣␣␣␣␣␣3.8051971560
H2␣␣␣␣␣␣␣␣␣␣␣␣-0.7745261239␣␣␣␣␣␣␣-0.2775291936␣␣␣␣␣␣␣␣4.3506967746
H3␣␣␣␣␣␣␣␣␣␣␣␣-1.9256618348␣␣␣␣␣␣␣␣0.2927045555␣␣␣␣␣␣␣-2.1370156139
End␣of␣Basis
Basis␣set
O.cc-pVDZ....
O1␣␣␣␣␣␣␣␣␣␣␣␣␣0.2162486684␣␣␣␣␣␣␣␣0.2196587542␣␣␣␣␣␣␣-2.9675781183
O2␣␣␣␣␣␣␣␣␣␣␣␣␣2.8171388123␣␣␣␣␣␣␣-0.2187115071␣␣␣␣␣␣␣␣0.3719375423
End␣of␣Basis
End␣of␣input
␣&SCF␣&End
Charge
-1.0
End␣of␣Input
␣&McKinley␣&End
Perturbation
Hessian
End␣of␣Input
```

From the output of the MCLR code

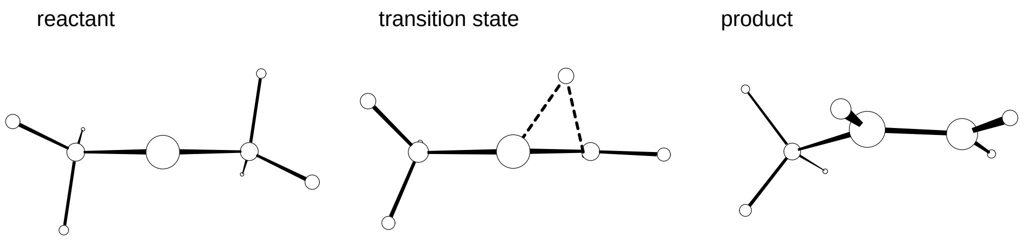
```
      ***********************************
      *                                 *
      * Harmonic frequencies in cm-1    *
      * Intensities in km/mole          *
      *                                 *
      * No correction due to curvlinear *
      * representations has been done   *
      *                                 *
      ***********************************
    Symmetry a
   ==============
                         1         2         3         4         5         6
            Freq.   i2027.40     i2.00     i0.07      0.05      0.07      2.02
...
...
                         7         8         9        10        11        12
            Freq.       3.57    145.36    278.41    574.44    675.27    759.94
...
...
                        13        14        15        16        17        18
            Freq.     927.78    943.60   1000.07   1225.34   1265.63   1442.57
...
...
                        19        20        21        22        23        24
            Freq.    1517.91   1800.86   1878.11   2294.83   3198.94   3262.66
```

we can conclude that we have one imaginary eigenvalue (modes 2-7 corresponds to the translational and rotational zero frequency modes) and that the structure found with this procedure indeed is a transition state. A post calculation analysis of the vibrational modes using the MOLDEN package confirm that the vibrational mode with the imaginary frequency is a mode which moves the proton from the oxygen to the carbon.

5.4 High quality wave functions at optimized structures

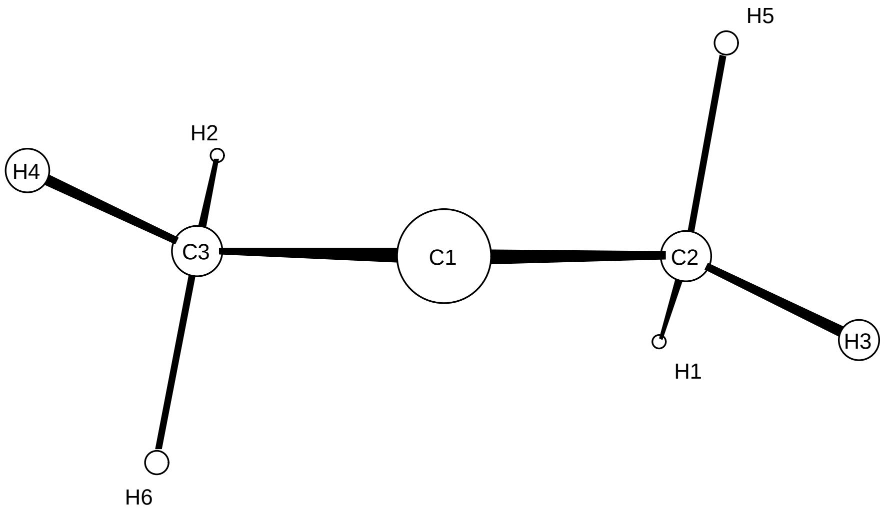
Here we will give an example of how geometrical structures obtained at one level of theory can be used in an analysis at high quality wave functions. Table 5.9 compiles the obtained CASSCF geometries for the dimethylcarbene to propene reaction (see Fig 5.8). They can be compared to the MP2 geometries [37]. The overall agreement is good.

Figure 5.8: Dimethylcarbene to propene reaction path



The wave function at each of the geometries was proved to be almost a single configuration. The second configuration in all the cases contributed by less than 5% to the weight of the wave function. It is a double excited replacement. Therefore, although MP2 is not generally expected to describe properly a bond formation in this case its behavior seems to be validated. The larger discrepancies appear in the carbon-carbon distances in the dimethylcarbene and in the transition state. On one hand the basis set used in the present example were small; on the other hand there are indications that the MP2 method overestimates the hyper conjugation effects present in the dimethylcarbene [37]. Figure 5.9 displays the dimethylcarbene with indication of the employed labeling.

Figure 5.9: Dimethylcarbene atom labeling



The main structural effects occurring during the reaction can be observed displayed in Table 5.9. As the rearrangement starts out one hydrogen atom (H_5) moves in a plane almost perpendicular to the plane formed by the three carbon atoms while the remaining two hydrogen atoms on the same methyl group swing very rapidly into a nearly planar position (see Figure 5.8 on page 161). As the π bond is formed we observe a contraction of the C_1-C_2 distance. In contrast, the spectator methyl group behaves as a rigid body. Their parameters were not compiled here but it rotates and bends slightly [37]. Focusing on the second half reaction, the moving hydrogen atom rotates into the plane of the carbon atoms to form the new C_1-H_5 bond. This movement is followed by a further shortening of the preformed C_1-C_2 bond, which acquires the bond distance of a typical double carbon bond, and smaller adjustments in the positions of the other atoms. The structures of the reactant, transition state, and product are shown in Figure 5.8.

As was already mentioned we will apply now higher-correlated methods for the reactant, product, and transition state system at the CASSCF optimized geometries to account for more accurate relative energies. In any case a small basis set has been used and therefore the goal is not to be extremely accurate. For more complete results see Ref. [37]. We are going to perform calculations with the MP2, MRCI, ACPF, CASPT2, CCSD, and CCSD(T) methods.

Starting with dimethylcarbene, we will use the following input file together with the `AUTO` program.

```
␣&SEWARD␣&END
Title
```

Table 5.9: Bond distances (Å) and bond angles (deg) of dimethylcarbene, propene, and their transition state[a]

	C_1C_3	C_1C_2	$C_2C_1C_3$	$C_1C_3H_6$	$C_2C_1C_3H_6$	C_2H_5	C_1H_5	$C_1C_2H_5$	$C_3C_1C_2H_5$
					Dimethylcarbene				
CAS[b]	1.497	1.497	110.9	102.9	88.9	1.099		102.9	88.9
MP2[c]	1.480	1.480	110.3	98.0	85.5	1.106		98.0	85.5
					Transition structure				
CAS[b]	1.512	1.394	114.6	106.1	68.6	1.287	1.315	58.6	76.6
MP2[c]	1.509	1.402	112.3	105.1	69.2	1.251	1.326	59.6	77.7
					Propene				
CAS[b]	1.505	1.344	124.9	110.7	59.4				
MP2[c]	1.501	1.338	124.4	111.1	59.4				

[a] C_1, carbenoid center; C_2, carbon which looses the hydrogen H_5. See Figure 5.9.

[b] Present results. CASSCF, ANO-S C 3*s*2*p*1*d*, H 2*d*1*p*. Two electrons in two orbitals.

[c] MP2 6-31G(2p,d), Ref. [37].

```
␣Dimethylcarbene␣singlet␣C2-sym
␣CASSCF(ANO-VDZP)␣opt␣geometry
Symmetry
␣XY
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣1.2019871414
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0369055124␣␣␣␣␣␣␣␣2.3301037548␣␣␣␣␣␣␣␣-.4006974719
End␣of␣basis
Basis␣set
H.ANO-S...2s1p.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣-.8322309260␣␣␣␣␣␣␣␣2.1305589948␣␣␣␣␣␣␣-2.2666729831
H2␣␣␣␣␣␣␣␣␣␣␣␣␣-.7079699536␣␣␣␣␣␣␣␣3.9796589218␣␣␣␣␣␣␣␣␣.5772009623
H3␣␣␣␣␣␣␣␣␣␣␣␣␣2.0671154914␣␣␣␣␣␣␣␣2.6585385786␣␣␣␣␣␣␣␣-.6954193494
End␣of␣basis
PkThrs
␣1.0E-10
End␣of␣input
␣&SCF␣&END
Title
Dmc
Occupied
7␣5
End␣of␣input
␣&RASSCF␣&END
Title
Dmc
Symmetry
␣1
Spin
␣1
Nactel
␣2␣␣0␣␣0
Inactive
```

```
␣6␣␣5
Ras2
␣1␣␣1
Thrs
1.0E-05,1.0E-03,1.0E-03
Iteration
50,25
LumOrb
End␣of␣input
␣&CASPT2␣&END
Title
Dmc
LRoot
1
Frozen
␣2␣␣1
End␣of␣input
␣&MOTRA␣&END
Title
Dmc
Frozen
␣2␣␣1
JobIph
End␣of␣input
␣&GUGA␣&END
Title
Dmc
Electrons
18
Spin
␣1
Inactive
␣4␣␣4
Active
␣1␣␣1
Ciall
␣1
Print
␣5
End␣of␣input
␣&MRCI␣&END
Title
Dimethylcarbene
SDCI
End␣of␣input
␣&MRCI␣&END
Title
Dimethylcarbene
ACPF
End␣of␣input
*␣Now␣we␣generate␣the␣single␣ref.␣function
*␣for␣coupled-cluster␣calculations
␣&RASSCF␣&END
Title
Dmc
Symmetry
␣1
Spin
␣1
Nactel
␣0␣␣0␣␣0
Inactive
␣7␣␣5
Ras2
␣0␣␣0
Thrs
```

```
1.0E-05,1.0E-03,1.0E-03
Iteration
50,25
LumOrb
OutOrbitals
␣Canonical
End␣of␣input
␣&MOTRA␣&END
Title
Dmc
Frozen
␣2␣␣1
JobIph
End␣of␣input
␣&CCSDT␣&END
Title
␣Dmc
CCT
Iterations
␣40
Triples
␣2
End␣of␣input
```

To run `AUTO` we will use the script:

```
#!/bin/ksh
export Project=dmc
export HomeDir=$PWD
export WorkDir=/temp/$LOGNAME/$Project
mkdir $WorkDir
molcas $HomeDir/$Project.input
rm -r $WorkDir
exit
```

Observe in the previous input that we have generated a multiconfigurational wave function for CASPT2, MRCI, and ACPF wave functions but a single configuration reference wave function (using `RASSCF` program with the options OUTOrbitals and CANOnical) for the CCSD and CCSD(T) wave functions. Notice also that to compute a multiconfigurational ACPF wave function we have to use the `MRCI` program, not the `CPF` module which does not accept more than one single reference. In all the highly correlated methods we have frozen the three carbon core orbitals because of the reasons already explained in section 5.1. For MRCI, ACPF, CCSD, and CCSD(T) the freezing is performed in the `MOTRA` step.

One question that can be addressed is which is the proper reference space for the multiconfigurational calculations. As was explained when we selected the active space for the geometry optimizations, we performed several tests at different stages in the reaction path and observed that the smallest meaningful active space, two electrons in two orbitals, was sufficient in all the cases. We can come back to this problem here to select the reference for CASPT2, MRCI, and ACPF methods. The simple analysis of the SCF orbital energies shows that in dimethylcarbene, for instance, the orbital energies of the C-H bonds are close to those of the C-C σ bonds and additionally those orbitals are strongly mixed along the reaction path. A balanced active space including all orbitals necessary to describe the shifting H-atom properly would require a full valence space of 18 electrons in 18 orbitals. This is not a feasible space, therefore we proceed with the minimal active space and analyze later the quality of the results. The CASSCF wave function will then include for dimethylcarbene and the transition state structure the $(\sigma)^2(\pi)^0$ and $(\sigma)^0(\pi)^2$ configurations correlating the non-bonded electrons localized at the carbenoid center where as for propene the active space include the equivalent valence π space.

The `GUGA` input must be built carefully. There are several ways to specify the reference configurations for the following methods. First, the keyword ELECTRONS refers to the total number of electrons that are going to be correlated, that is, all except those frozen in the previous `MOTRA` step. Keywords INACTIVE and ACTIVE are optional and describe the number of inactive (occupation two in all the reference configurations) and active (varying occupation number in the reference configurations) orbitals of the space. Here ACTIVE indicates one orbital of each of the symmetries. The following keyword CIALL indicates that the reference space will be the full CI within the subspace of active orbitals. It must be always followed by symmetry index (number of the irrep) for the resulting wave function, one here.

For the transition state structure we do not impose any symmetry restriction, therefore the calculations are performed in the C_1 group with the input file:

```
␣&SEWARD␣&END
Title
␣Dimethylcarbene␣to␣propene
␣Transition␣State␣C1␣symmetry
␣CASSCF␣(ANO-VDZP)␣opt␣geometry
Basis␣set
C.ANO-S...3s2p1d.
End␣of␣basis
Basis␣set
H.ANO-S...2s1p.
End␣of␣basis
PkThrs
␣1.0E-10
End␣of␣input
␣&SCF␣&END
Title
␣Ts
Occupied
␣12
End␣of␣input
␣&MBPT2␣&END
Title
␣Ts
Frozen
␣3
End␣of␣input
␣&RASSCF␣&END
Title
␣Ts
Symmetry
␣1
Spin
␣1
Nactel
␣2␣␣0␣␣0
Inactive
␣11
Ras2
␣2
Iteration
50,25
LumOrb
End␣of␣input
␣&CASPT2␣&END
Title
␣Ts
LRoot
```

```
␣1
Frozen
␣3
End␣of␣input
␣&MOTRA␣&END
Title
␣Ts
Frozen
␣3
JobIph
End␣of␣input
␣&GUGA␣&END
Title
␣Ts
Electrons
␣18
Spin
␣1
Inactive
␣8
Active
␣2
Ciall
␣1
Print
␣5
End␣of␣input
␣&MRCI␣&END
Title
␣Ts
SDCI
End␣of␣input
␣&MRCI␣&END
Title
␣Ts
ACPF
End␣of␣input
␣&RASSCF␣&END
Title
␣Ts
Symmetry
␣1
Spin
␣1
Nactel
␣0␣␣0␣␣0
Inactive
␣12
Ras2
␣0
Iteration
50,25
LumOrb
OutOrbitals
␣Canonical
End␣of␣input
␣&MOTRA␣&END
Title
␣Ts
Frozen
␣3
JobIph
End␣of␣input
␣&CCSDT␣&END
Title
␣Ts
```

```
CCT
Iterations
␣40
Triples
␣2
End␣of␣input
```

Finally we compute the wave functions for the product, propene, in the C$_s$ symmetry group with the input:

```
␣&SEWARD␣&END
Title
␣Propene␣singlet␣Cs-sym
␣CASSCF(ANO-VDZP)␣opt␣geometry
Symmetry
␣Z
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣-2.4150580342␣␣␣␣␣␣␣␣␣.2276105054␣␣␣␣␣␣␣␣␣.0000000000
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0418519070␣␣␣␣␣␣␣␣␣.8733601069␣␣␣␣␣␣␣␣␣.0000000000
C3␣␣␣␣␣␣␣␣␣␣␣␣␣2.2070668305␣␣␣␣␣␣␣␣-.9719171861␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
Basis␣set
H.ANO-S...2s1p.
H1␣␣␣␣␣␣␣␣␣␣␣␣-3.0022907382␣␣␣␣␣␣␣-1.7332097498␣␣␣␣␣␣␣␣␣.0000000000
H2␣␣␣␣␣␣␣␣␣␣␣␣-3.8884900111␣␣␣␣␣␣␣␣1.6454331428␣␣␣␣␣␣␣␣␣.0000000000
H3␣␣␣␣␣␣␣␣␣␣␣␣␣␣.5407865292␣␣␣␣␣␣␣␣2.8637419734␣␣␣␣␣␣␣␣␣.0000000000
H4␣␣␣␣␣␣␣␣␣␣␣␣␣1.5296107561␣␣␣␣␣␣␣-2.9154199848␣␣␣␣␣␣␣␣␣.0000000000
H5␣␣␣␣␣␣␣␣␣␣␣␣␣3.3992878183␣␣␣␣␣␣␣␣-.6985812202␣␣␣␣␣␣␣␣1.6621549148
End␣of␣basis
PkThrs
␣1.0E-10
End␣of␣input
␣&SCF␣&END
Title
Propene
Occupied
10␣2
End␣of␣input
␣&MBPT2␣&END
Title
␣Propene
Frozen
␣3␣0
End␣of␣input
␣&RASSCF␣&END
Title
Propene
Symmetry
1
Spin
1
Nactel
␣2␣␣0␣␣0
Inactive
10␣␣1
Ras2
␣0␣␣2
Thrs
1.0E-05,1.0E-03,1.0E-03
Iteration
50,25
LumOrb
End␣of␣input
␣&CASPT2␣&END
```

```
Title
Propene
LRoot
1
Frozen
␣3␣␣0
End␣of␣input
␣&MOTRA␣&END
Title
Propene
Frozen
␣3␣␣0
JobIph
End␣of␣input
␣&GUGA␣&END
Title
Propene
Electrons
18
Spin
␣1
Inactive
␣7␣␣1
Active
␣0␣␣2
Ciall
␣1
Print
␣5
End␣of␣input
␣&MRCI␣&END
Title
Propene
SDCI
End␣of␣input
␣&MRCI␣&END
Title
Propene
ACPF
End␣of␣input
␣&RASSCF␣&END
Title
Propene
Symmetry
1
Spin
1
Nactel
␣0␣␣0␣␣0
Inactive
10␣␣2
Ras2
␣0␣␣0
Thrs
1.0E-05,1.0E-03,1.0E-03
Iteration
50,25
LumOrb
OutOrbitals
␣Canonical
End␣of␣input
␣&MOTRA␣&END
Title
Propene
Frozen
␣3␣␣0
```

```
JobIph
End␣of␣input
␣&CCSDT␣&END
Title
␣Propene
CCT
Iterations
␣40
Triples
␣2
End␣of␣input
```

Table 5.10 compiles the total and relative energies obtained for the studied reaction at the different levels of theory employed.

We can discuss now the quality of the results obtained and their reliability (for a more careful discussion of the accuracy of quantum chemical calculations see Ref. [25]). In first place we have to consider that a valence double-zeta plus polarization basis set is somewhat small to obtain accurate results. At least a triple-zeta quality would be required. The present results have, however, the goal to serve as an example. We already pointed out that the CASSCF geometries were very similar to the MP2 reported geometries [37]. This fact validates both methods. MP2 provides remarkably accurate geometries using basis sets of triple-zeta quality, as in Ref. [37], in situations were the systems can be described as singly configurational, as the CASSCF calculations show. The Hartree-Fock configuration has a contribution of more than 95% in all three structures, while the largest weight for another configuration appears in propene for $(\pi)^0(\pi^*)^2$ (4.2%).

The MRCI calculations provide also one test of the validity of the reference wave function. For instance, the MRCI output for propene is:

```
              FINAL RESULTS FOR STATE NR   1
CORRESPONDING ROOT OF REFERENCE CI IS NR:  1
           REFERENCE CI ENERGY: -117.12226386
        EXTRA-REFERENCE WEIGHT:      .11847074
         CI CORRELATION ENERGY:     -.38063043
                     CI ENERGY: -117.50289429
           DAVIDSON CORRECTION:     -.05115380
              CORRECTED ENERGY: -117.55404809
               ACPF CORRECTION:     -.04480105
              CORRECTED ENERGY: -117.54769535
     CI-COEFFICIENTS LARGER THAN  .050
 NOTE: THE FOLLOWING ORBITALS WERE FROZEN
 ALREADY AT THE INTEGRAL TRANSFORMATION STEP
 AND DO NOT EXPLICITLY APPEAR:
       SYMMETRY:   1   2
     PRE-FROZEN:   3   0
 ORDER OF SPIN-COUPLING: (PRE-FROZEN, NOT SHOWN)
                         (FROZEN, NOT SHOWN)
                          VIRTUAL
                          ADDED VALENCE
                          INACTIVE
                          ACTIVE
 ORBITALS ARE NUMBERED WITHIN EACH SEPARATE SYMMETRY.
     CONFIGURATION     32   COEFFICIENT  -.165909   REFERENCE
SYMMETRY             1  1  1  1  1  1  1  2  2  2
ORBITALS             4  5  6  7  8  9 10  1  2  3
OCCUPATION           2  2  2  2  2  2  2  2  0  2
SPIN-COUPLING        3  3  3  3  3  3  3  3  0  3
     CONFIGURATION     33   COEFFICIENT  -.000370   REFERENCE
SYMMETRY             1  1  1  1  1  1  1  2  2  2
ORBITALS             4  5  6  7  8  9 10  1  2  3
```

Table 5.10: Total (au) and relative (Kcal/mol, in braces) energies obtained at the different theory levels for the reaction path from dimethylcarbene to propene

Single configurational methods

method	RHF	MP2	CCSD	CCSD(T)
Dimethylcarbene				
	-117.001170	-117.392130	-117.442422	-117.455788
Transition state structure				
	-116.972670	-117.381342	-117.424088	-117.439239
BH[a]	(17.88)	(6.77)	(11.50)	(10.38)
Propene				
	-117.094700	-117.504053	-117.545133	-117.559729
EX[b]	(-58.69)	(-70.23)	(-64.45)	(-65.22)

Multiconfigurational methods

method	CASSCF	CASPT2	SD-MRCI+Q	ACPF
Dimethylcarbene				
	-117.020462	-117.398025	-117.447395	-117.448813
Transition state structure				
	-116.988419	-117.383017	-117.430951	-117.432554
BH[a]	(20.11)	(9.42)	(10.32)	(10.20)
Propene				
	-117.122264	-117.506315	-117.554048	-117.554874
EX[b]	(-63.88)	(-67.95)	(-66.93)	(-66.55)

[a] Barrier height. Needs to be corrected with the zero point vibrational correction.

[b] Exothermicity. Needs to be corrected with the zero point vibrational correction.

```
  OCCUPATION          2  2  2  2  2  2  2  2  1  1
  SPIN-COUPLING       3  3  3  3  3  3  3  3  1  2
       CONFIGURATION     34    COEFFICIENT    .924123   REFERENCE
  SYMMETRY            1  1  1  1  1  1  1  2  2  2
  ORBITALS            4  5  6  7  8  9 10  1  2  3
  OCCUPATION          2  2  2  2  2  2  2  2  2  0
  SPIN-COUPLING       3  3  3  3  3  3  3  3  3  0
 **************************************************************
```

The Hartree-Fock configuration contributes to the MRCI configuration with a weight of 85.4%, while the next configuration contributes by 2.8%. Similar conclusions can be obtained analyzing the ACPF results and for the other structures. We will keep the MRCI results including the Davidson correction (MRCI+Q) which corrects for the size-inconsistency of the truncated CI expansion [25].

For CASPT2 the evaluation criteria were already commented in section 5.5. The portion of the `CASPT2` output for propene is:

```
       Reference energy:        -117.1222638304
       E2 (Non-variational):       -.3851719971
       E2 (Variational):           -.3840516039
       Total energy:            -117.5063154343
       Residual norm:               .0000000000
       Reference weight:            .87905
       Contributions to the CASPT2 correlation energy
       Active & Virtual Only:          -.0057016698
       One Inactive Excited:           -.0828133881
       Two Inactive Excited:           -.2966569393
 ----------------------------------------------------------------------------
 Report on small energy denominators, large components, and large energy contributions.
 The ACTIVE-MIX index denotes linear combinations which gives ON expansion functions
   and makes H0 diagonal within type.
 DENOMINATOR: The (H0_ii - E0) value from the above-mentioned diagonal approximation.
 RHS value: Right-Hand Side of CASPT2 Eqs.
 COEFFICIENT: Multiplies each of the above ON terms in the first-order wave function.
 Thresholds used:
         Denominators:  .3000
           Components:  .0250
 Energy contributions:  .0050
 CASE SYMM ACTIVE  NON-ACT IND    DENOMINATOR  RHS VALUE  COEFFICIENT CONTRIBUTION
 AIVX  1  Mu1.0003 In1.004 Se1.022  2.28926570 .05988708  -.02615995  -.00156664
```

The weight of the CASSCF reference to the first-order wave function is here 87.9%, very close to the weights obtained for the dimethylcarbene and the transition state structure, and there is only a small contribution to the wave function and energy which is larger than the selected thresholds. This should not be considered as a intruder state, but as a contribution from the fourth inactive orbital which could be, eventually, included in the active space. The contribution to the second-order energy in this case is smaller than 1 Kcal/mol. It can be observed that the same contribution shows up for the transition state structure but not for the dimethylcarbene. In principle this could be an indication that a larger active space, that is, four electrons in four orbitals, would give a slightly more accurate CASPT2 energy. The present results will probably overestimate the second-order energies for the transition state structure and the propene, leading to a slightly smaller activation barrier and a slightly larger exothermicity, as can be observed in Table 5.10. The orbitals pointed out as responsible for the large contributions in propene are the fourth inactive and 22nd secondary orbitals of the first symmetry. They are too deep and too high, respectively, to expect that an increase in the active space could in fact represent a great improvement in the CASPT2 result. In any case we tested for four orbitals-four electrons CASSCF/CASPT2 calculations and the results were very similar to those presented here.

Finally we can analyze the so-called τ_1-diagnostic [38] for the coupled-cluster wave functions. τ_1 is defined for closed-shell coupled-cluster methods as the Euclidian norm of the vector of T_1 amplitudes normalized by the number of electrons correlated: $\tau_1 = ||T_1||/N_{el}^{1/2}$. In the output of the `CCSD` program we have:

```
       Convergence after  17  Iterations
       Total energy (diff) :    -117.54513288       -.00000061
       Correlation energy  :        -.45043295
       E1aa   contribution :         .00000000
       E1bb   contribution :         .00000000
       E2aaaa contribution :        -.04300448
       E2bbbb contribution :        -.04300448
       E2abab contribution :        -.36442400
  Five largest amplitudes of :T1aa
   SYMA   SYMB   SYMI   SYMJ     A      B      I      J      VALUE
     2      0      2      0      4      0      2      0     -.0149364994
     2      0      2      0      2      0      2      0      .0132231037
     2      0      2      0      8      0      2      0     -.0104167047
     2      0      2      0      7      0      2      0     -.0103366543
     2      0      2      0      1      0      2      0      .0077537734
  Euclidian norm is :      .0403635306
  Five largest amplitudes of :T1bb
   SYMA   SYMB   SYMI   SYMJ     A      B      I      J      VALUE
     2      0      2      0      4      0      2      0     -.0149364994
     2      0      2      0      2      0      2      0      .0132231037
     2      0      2      0      8      0      2      0     -.0104167047
     2      0      2      0      7      0      2      0     -.0103366543
     2      0      2      0      1      0      2      0      .0077537734
  Euclidian norm is :      .0403635306
```

In this case T1aa and T1bb are identical because we are computing a closed-shell singlet state. The five largest T_1 amplitudes are printed, as well as the Euclidian norm. Here the number of correlated electrons is 18, therefore the value for the τ_1 diagnostic is 0.01. This value can be considered acceptable as evaluation of the quality of the calculation. The use of τ_1 as a diagnostic is based on an observed empirical correlation: larger values give poor CCSD results for molecular structures, binding energies, and vibrational frequencies [39]. It was considered that values larger than 0.02 indicated that results from single-reference electron correlation methods limited to single and double excitations should be viewed with caution.

There are several considerations concerning the τ_1 diagnostic [38]. First, it is only valid within the frozen core approximation and it was defined for coupled-cluster procedures using SCF molecular orbitals in the reference function. Second, it is a measure of the importance of non-dynamical electron correlation effects and not of the degree of the multireference effects. Sometimes the two effects are related, but not always (see discussion in Ref. [39]). Finally, the performance of the CCSD(T) method is reasonably good even in situations where τ_1 has a value as large as 0.08. In conclusion, the use of τ_1 together with other wave function analysis, such as explicitely examining the largest T_1 and T_2 amplitudes, is the best approach to evaluate the quality of the calculations but this must be done with extreme caution.

As the present systems are reasonably well described by a single determinant reference function there is no doubt that the CCSD(T) method provides the most accurate results. Here CASPT2, MRCI+Q, ACPF, and CCSD(T) predict the barrier height from the reactant to the transition state with an accuracy better than 1 Kcal/mol. The correspondence is somewhat worse, about 3 Kcal/mol, for the exothermicity. As the difference is largest for the

CCSD(T) method we may conclude than triple and higher order excitations are of importance to achieve a balanced correlation treatment, in particular with respect to the partially occupied π^* orbital at the carbenoid center. It is also noticeable that the relative MP2 energies appear to be shifted about 3-4 Kcal/mol towards lower values. This effect may be due to the overestimation of the hyper-conjugation effect which appears to be strongest in dimethylcarbene [40, 37].

Additional factors affecting the accuracy of the results obtained are the zero point vibrational energy correction and, of course, the saturation of the one particle basis sets. The zero point vibrational correction could be computed by performing a numerical harmonic vibrational analysis at the CASSCF level using *MOLCAS*. At the MP2 level [37] the obtained values were -1.1 Kcal/mol and 2.4 Kcal/mol for the activation barrier height and exothermicity, respectively. Therefore, if we take as our best values the CCSD(T) results of 10.4 and -65.2 Kcal/mol, respectively, our prediction would be an activation barrier height of 9.3 Kcal/mol and an exothermicity of -62.8 Kcal/mol. Calculations with larger basis sets and MP2 geometries gave 7.4 and -66.2 Kcal/mol, respectively [37]. The experimental estimation gives a lower limit to the activation barrier of 3.3 Kcal/mol [37].

MOLCAS provides also a number of one-electron properties which can be useful to analyze the chemical behavior of the systems. For instance, the Mulliken population analysis is available for the RHF, CASSCF, CASPT2, MRCI, and ACPF wave functions. Mulliken charges are known to be strongly biased by the choice of the basis sets, nevertheless one can restrict the analysis to the relative charge differences during the course of the reaction to obtain a qualitative picture. We can use, for instance, the charge distribution obtained for the MRCI wave function, which is listed in Table 5.11. Take into account that the absolute values of the charges can vary with the change of basis set.

Table 5.11: Mulliken's population analysis (partial charges) for the reaction path from dimethylcarbene to propene. MRCI wave functions.

C_2^a	C_1^b	H_5^c	Σ^d	$H_1+H_3^e$	Me^f
		Dimethylcarbene			
-0.12	-0.13	0.05	-0.20	0.14	0.07
		Transition state structure			
-0.02	-0.23	0.05	-0.20	0.17	0.02
		Propene			
-0.18	-0.02	0.05	-0.15	0.18	-0.02

[a] Carbon from which the hydrogen is withdrawn.

[b] Central carbenoid carbon.

[c] Migrating hydrogen.

[d] Sum of charges for centers C_2, C_1, and H_5.

[e] Sum of charges for the remaining hydrogens attached to C_2.

[f] Sum of charges for the spectator methyl group.

In dimethylcarbene both the medium and terminal carbons appear equally charged. During the migration of hydrogen H_5 charge flows from the hydrogen donating carbon, C_2, to the carbenoid center. For the second half of the reaction the charge flows back to the terminal carbon from the centered carbon, probably due to the effect of the π delocalization.

5.5 Excited states.

The accurate calculation of excited electronic states has been a challenge for quantum chemistry. The possibility for accurate calculations of such states in molecules has only recently been made possible through the development of new quantum chemical techniques. CASPT2 is currently one of the more successful methods to compute excited states due to its balance between accuracy and cost. In addition to the intrinsic limitations of the method, photochemistry and photophysics involves a large number of situations and mechanisms which complicate the problems enormously. In the present section we are going to show a systematic way to deal with a large number of states in a molecule. We have selected the thiophene molecule and our goal will be to compute the lowest valence and Rydberg singlet states at the ground state geometry. This can be considered to be the gas-phase absorption spectrum of the molecule. The calculations comprise an extensive use of the RASSCF, CASPT2, and RASSI programs. Selection of proper active spaces, building of appropriate diffuse basis functions, calculation of transition dipole moments, and use of the level-shift technique in CASPT2 will be some of the topics covered.

5.5.1 The vertical spectrum of thiophene.

Besides the usual limitation typical of any *ab initio* procedure due to the size of the system and the calculation of the integrals, the CASPT2 method has the basic limitation of the size and selection of the active space in the preliminary CASSCF step, not only because the space cannot be too large but because the active space defines the type and number of configurations (read excitations) to be included in the multiconfigurational wave functions. The near-degenerate configurations describing all states must be present in the reference wave function. Therefore, certain knowledge of the system is necessary to design the calculation and, for excited states, this will limit the number of states we are able to study.

Planning the calculations.

Thiophene is a planar five membered ring molecule containing one sulfur and four carbon atoms. The π structure of the system contains two conjugated double bonds between carbon atoms. Therefore, the orbital π valence structure is composed by two π bonding, two π^* antibonding orbitals, and one π nonbonding orbital placed on the sulfur atom. The π orbitals are the highest occupied ones in this type of systems and excitations from them form the UV spectrum in gas phase and solution. Also, typical orbitals involved in low-lying excited states are the lone-pair orbitals such as the sulfur n orbital co-planar with the σ skeleton of the molecule. On the other hand, σ orbitals forming C–H and C–C bonds do not participate in the low-lying excited electronic states. One has, however to be careful here. In thiophene there are low-lying virtual σ that give rise to excited states in the region around 6 eV [41].

With this in mind we have to include at least the three π and two π^* valence orbitals and the valence σ lone-pair on the sulfur in the active space. The molecule belongs to the C_{2v} point group, therefore we have three b_1 and two a_2 π, π^* orbitals and one a_1 n orbital. That is, our minimal valence active space can be labeled (1302), where each number corresponds to the number of a_1,b_1, b_2, and a_2 orbitals, respectively.

But the valence states are not the only states present at low energies. In a gas-phase spectrum of a neutral molecule the Rydberg states start to appear at energies above 5 eV. Therefore, they must be simultaneously included in the calculations. The Rydberg orbitals are large compared to the molecular dimension and therefore have quasi atomic shapes. Rydberg states are commonly labeled as excited states of atoms with a principal quantum number n and the usual angular quantum numbers l and m. For molecules containing only first row atoms n conventionally starts with 3. This convention is actually used also in a molecule like thiophene, although in sulfur the valence electrons are in the third shell. Increasing the value of n will lead to more and more diffuse orbitals, eventually converging to an ionized state of the molecule. The lowest Rydberg state corresponds to the excitation HOMO→$3s$. The next components will be $3p_x$, $3p_y$, and $3p_z$, followed by the five components of $3d$.

The Rydberg orbitals classify into the point group like their corresponding atomic orbitals. Therefore, a look at the character table (see Table 5.2) indicates that in C_{2v} the s,p_z,d_{z^2}, and $d_{x^2-y^2}$ Rydberg orbitals belong to symmetry a_1, p_x and d_{xz} to symmetry b_1, p_y and d_{yz} to symmetry b_2 and, finally, d_{xy} to symmetry a_2. According to the labeling defined above the nine lowest Rydberg orbitals classify to (4221). It is obvious that we cannot normally afford to have simultaneously the whole valence plus Rydberg space (15 active orbitals in the present example). Therefore we are going to exploit the symmetry properties to select different active spaces.

By inspection of the SCF orbital energies or the ionization potentials of the molecule we observe that the highest occupied orbitals HOMO ($1\mathrm{a}_2$) and HOMO-1 ($2\mathrm{b}_1$) are reasonably close in energy (around 0.6 eV). Therefore, two Rydberg series close in energy can be expected at low energies, the first one arising from the HOMO orbital and the second from the HOMO-1 orbital. By exciting one electron from each of those orbitals to each one of the Rydberg orbitals we know the symmetry of the resulting state. For instance, the excitation HOMO (a_2) → 3s (a_1) leads to a A_2 by direct product of the symmetry representations. Table 5.12 contains the analysis for the Rydberg states arising both from HOMO and HOMO-1 orbitals to the n=3 Rydberg orbitals. They form the two lowest Rydberg series. We want also to locate the state from the lone-pair HOMO-2 ($11\mathrm{a}_1$) to 3s.

The computed states will use different partitionings of the active space. The basic valence space (1302) must be included in all the cases. The valence $\pi \rightarrow \pi^*$ states only involve excitations into the π and π^* orbitals. Therefore they belong to the A_1 and B_2 symmetries. In addition we can have single excitations (Rydberg states) from the occupied π orbitals to the Rydberg orbitals of b_1 and a_2 symmetries. The number of Rydberg orbitals belonging to those symmetries is (0201). Thus, the final space to compute simultaneously valence and Rydberg $\pi \rightarrow \pi^*$ states is (1302) + (0201): (1503). The same space can be used to compute $n \rightarrow \pi^*$ states because the n orbital and the π^* orbitals are included into the active space. The symmetries of these states, however, will be A_2 and B_1. In the table we also have another division for the A_2 and B_1, $\pi \rightarrow$R(σ), and A_1, $n \rightarrow$R(σ), (only the $n \rightarrow 3s$) Rydberg states, using an active space (5322). We have, therefore, divided the excited states to be computed by symmetries and active space. State-average CASSCF calculations for each one of the cases have to be performed. The only question which remains is how many roots we have to include in each of the cases. This is also determined by the symmetry and active space available. For instance, for the $\pi \rightarrow \pi^*$ A_1 states, we want to compute the ground state plus three Rydberg states (see Table 5.12 in both HOMO and HOMO-1 → n=3 series) plus a certain number of valence states. If we do not have any previous experience we may think of three or four possible valence states but we know that the usual number of low-lying valence states is close to the number of valence singly excited states, in this case two of A_1 symmetry. This does not mean that the states are going to be described by one single configuration; it is simply an estimation of the number of relevant states based on

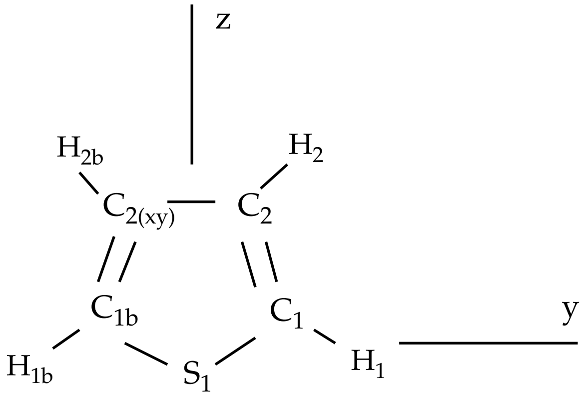


Figure 5.10: Thiophene

Table 5.12: Selection of active spaces in thiophene.

	Symmetries			
	a_1	b_1	b_2	a_2
Frozen orb.	5	1	3	0
Inactive orb.	6	0	4	0
Valence active orb.	1	3	0	2

Rydberg states

HOMO→n=3		State	HOMO-1→n=3		State	HOMO-2→n=3		State[a]
(π) a_2 →	3sa_1	A_2	(π) b_1 →	3sa_1	B_1	(n) a_1 →	3sa_1	A_1
	3pa_1	A_2		3pa_1	B_1			
	3pb_1	B_2		3pb_1	A_1			
	3pb_2	B_1		3pb_2	A_2			
	3da_1	A_2		3da_1	B_1			
	3da_1	A_2		3da_1	B_1			
	3db_1	B_2		3db_1	A_1			
	3db_2	B_1		3db_2	A_2			
	3da_2	A_1		3da_2	B_2			

Total active space

A_1, B_2 states ($\pi \rightarrow \pi^*$) $_1$, B_2 states ($\pi \rightarrow$R(π^*)) A_2, B_1 states ($n \rightarrow \pi^*$)	Valence (1302) + Rydberg (0201) = (1503)
A_2, B_1 states ($\pi \rightarrow$R(σ)) A_1 states ($n \rightarrow$R(σ))	Valence (1302) + Rydberg (4020) = (5322)

[a] Only considered up to the A_1 (3s) state because the remaining are expected at higher energy.

experience. In summary, we expect to compute six A_1 states and therefore we include six roots in the CASSCF state-average input.

It is not uncommon that one or more valence states do not appear in the initial CASSCF calculation including the desired roots and other higher Rydberg states. This is due to the fact that valence states usually require larger dynamical correlation corrections than the Rydberg states. Therefore in a CASSCF calculation the Rydberg states are, in general, lower in energy than the valence states. The dynamical correlation included by the CASPT2 method will place the states correctly. However this is only possible if the states are present in the CASSCF calculation. It is then necessary to be sure that the states are located at the CASSCF level. Maybe it is necessary to increase the number of roots and in special cases like those with low symmetry even to delete Rydberg orbitals from the active space [42, 43, 32, 44].

In the following we will describe briefly the calculations [45]. A detailed report of the vertical excited spectrum of thiophene can be found in references [45, 46]. The selection of the active spaces in that work included additional orbitals to minimize the effect of intruder states. The availability of the level-shift technique in later versions of MOLCAS allow us to use a smaller active space.

Generating Rydberg basis functions

First we describe a method for generating Rydberg basis functions for molecules. Such Rydberg orbitals are diffuse and thus require diffuse basis functions. Due to this diffuseness they are not "localized" to atoms in the sense that valence orbitals are, but should be considered to be spread out over the entire molecule.

The basis of the method lies in the fact that if we add an electron into a virtual orbital, the energy for the system is increased by the orbital energy, according to Koopmanns' theorem. The reorganizational effects are very minor for the diffuse virtual orbitals. Thus adding an electron into a virtual orbital for a cation is an reasonable approximation to the proper Rydberg state. A more extensive discussion of the method outlined below can be found in [43].

The method can be broken down into a few steps (see Ref. [43] for details):

1. Perform a RHF or valence CASSCF calculation of the system with one electron removed, using the `RASSCF` program. This will determine the center of charge which is a suitable choice to center the Rydberg basis function expansion. The result is rather insensitive to this choice.
2. Add a suitable diffuse primitive basis set at the center of charge. We use as universal exponents those optimized by Kaufmann *et al.* [47] for Rydberg wave functions.
3. Repeat the RHF or CASSCF calculation in the new basis.
4. Construct the basis set using the program GENANO and use the lowest virtual function to define the basis set.

It is better not to use an extremely large valence basis set to perform these calculations. The best choice is a double-zeta or double-zeta plus polarization basis set. In this example we will use benzene which have a natural origin in the center of the ring. Thus we have eliminated the step of determining the center of charge. Also we have made the simplification of only considering s-functions.

The procedure we will follow is

1. Create inputs for `SEWARD`, `SCF`, `RASSCF`, and `GENANO`.
2. Create a shell script to run `SEWARD`, `SCF`, and `RASSCF`, and run the job.
3. Hand edit the resulting formated orbital file, *`C6H6.RasOrb`*. Set the occupation numbers for the occupied space to zero, while the first three virtual orbitals in the first irreducible representation get the occupation numbers 10^{-1}, 10^{-2} and 10^{-3} respectively. These occupation numbers are quite arbitrary as long as they form a decreasing sequence.
4. Create a shell script to run `GENANO` and run the job.
5. The resulting file *`C6H6.Ano`* now contains the contraction coefficients. Merge this file with the exponents in the `SEWARD` input to obtain the final contracted basis set. We normally use only one function of each type.

The radial extent of the resulting basis functions is shown in figure 5.11.

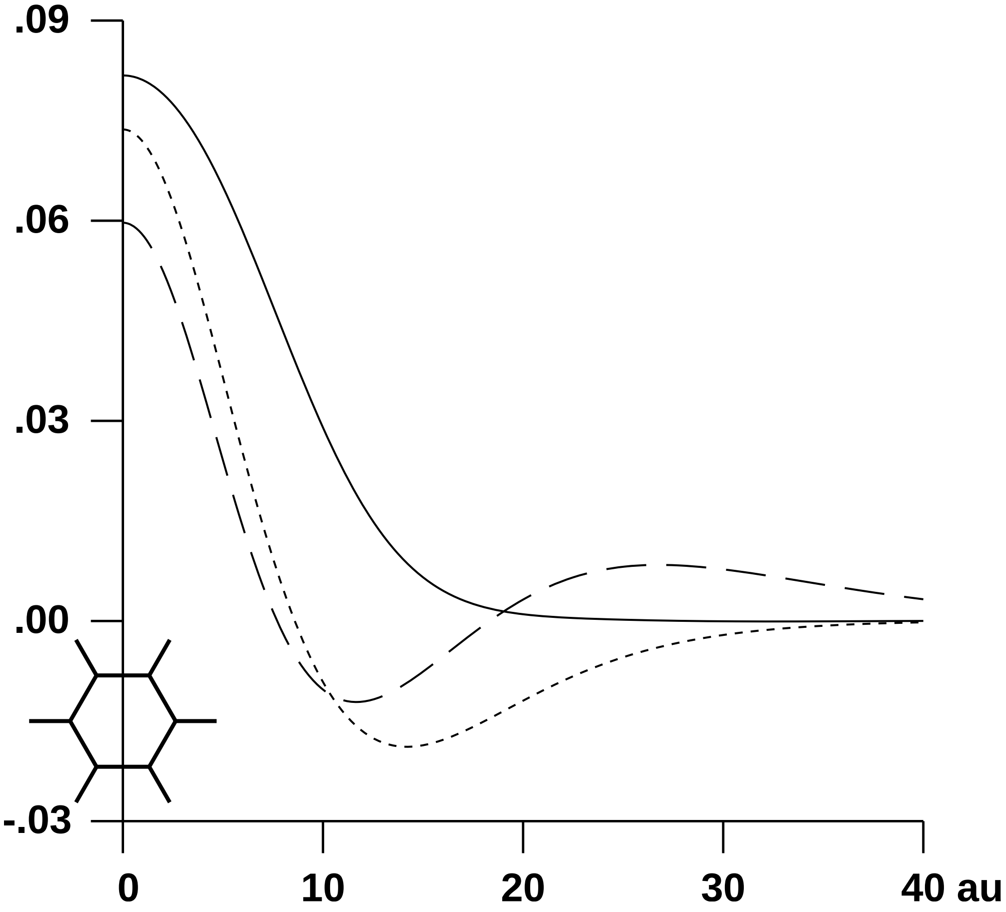


Figure 5.11: Radial extent of the Rydberg orbitals

Here are the inputs used for this example. First the SEWARD input using the uncontracted Rydberg functions (note that only the s-type Rydberg basis is shown).

```
␣&SEWARD␣&END
Title
␣Benzene␣molecule.
Symmetry
X␣Y␣Z
*OneOnly
Basis␣set
C.ano-s...3s2p1d.
C1␣␣␣␣2.636169␣␣␣␣␣.000000␣␣␣␣␣.000000
C2␣␣␣␣1.318084␣␣␣␣2.282990␣␣␣␣␣.000000
End␣of␣basis
Basis␣set
H.ano-s...2s1p.
H1␣␣␣␣4.684633␣␣␣␣␣.000000␣␣␣␣␣.000000
H2␣␣␣␣2.342316␣␣␣␣4.057011␣␣␣␣␣.000000
End␣of␣basis
Basis␣set
X....8s8p8d.␣/␣Inline
␣␣0.0␣0
8␣8
.02462393␣.01125334␣.00585838␣.00334597␣.00204842␣.00132364␣.00089310␣.00062431
1.0␣0.0␣0.0␣0.0␣0.0␣0.0␣0.0␣0.0
0.0␣1.0␣0.0␣0.0␣0.0␣0.0␣0.0␣0.0
0.0␣0.0␣1.0␣0.0␣0.0␣0.0␣0.0␣0.0
0.0␣0.0␣0.0␣1.0␣0.0␣0.0␣0.0␣0.0
0.0␣0.0␣0.0␣0.0␣1.0␣0.0␣0.0␣0.0
0.0␣0.0␣0.0␣0.0␣0.0␣1.0␣0.0␣0.0
0.0␣0.0␣0.0␣0.0␣0.0␣0.0␣1.0␣0.0
0.0␣0.0␣0.0␣0.0␣0.0␣0.0␣0.0␣1.0
X␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣␣.000000
End␣of␣basis
End␣of␣input
```

Once computed, the contracted functions will replace the uncontracted ones. In the usual calculations we are going to use one function of each type, $1s1p1d$, but we can keep three of them if we want to increase the Rydberg basis for some particular use. Here is the input listing for the generation of the ANO. Note that in newer versions of MOLCAS the sequence of calculations is driven by the input list. You can skip parts of the calculation by commenting out (with a *) the corresponding namelist input (for example * &SEWARD &END skips the integral calculation).

```
!ln␣-s␣C6H6.OneInt␣␣␣␣␣␣␣␣ONEINT
␣&SEWARD␣&END
Title
␣Benzene␣molecule.
Symmetry
X␣Y␣Z
*OneOnly
Basis␣set
C.ano-s...3s2p1d.
C1␣␣␣␣2.636169␣␣␣␣␣.000000␣␣␣␣␣.000000
C2␣␣␣␣1.318084␣␣␣␣2.282990␣␣␣␣␣.000000
End␣of␣basis
Basis␣set
H.ano-s...2s1p.
H1␣␣␣␣4.684633␣␣␣␣␣.000000␣␣␣␣␣.000000
H2␣␣␣␣2.342316␣␣␣␣4.057011␣␣␣␣␣.000000
End␣of␣basis
Basis␣set
X....1s1p1d.␣/␣Inline
```

```
␣␣0.0␣0
8␣1
.02462393␣.01125334␣.00585838␣.00334597␣.00204842␣.00132364␣.00089310␣.00062431
␣␣␣.15531366␣␣-.26126804␣␣␣.38654527
␣-1.53362747␣-1.27182240␣␣␣.94560891
␣␣1.10186802␣␣␣.95250581␣-1.24269525
␣-1.70918216␣␣␣.49632170␣-2.22724281
␣␣2.03031830␣␣␣.68292933␣␣1.94719179
␣-1.73187442␣␣-.56245782␣␣␣.68883478
␣␣␣.92694465␣␣␣.30675927␣␣␣.15138171
␣␣-.22934028␣␣-.07852136␣␣-.02092438
X␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣␣.000000
End␣of␣basis
␣&SCF␣&END
Title
␣Benzene␣molecule.
Occupied
␣6␣5␣4␣3␣1␣1␣1␣0
End␣of␣input
!ln␣-s␣$Home/C6H6.RasOrb␣␣RASORB
␣&RASSCF␣&END
Title
␣Benzene␣molecule
Symmetry
␣7
Spin
␣2
nActEl
␣␣1␣0␣0
Inactive
␣6␣5␣4␣3␣1␣1␣0␣0
Ras2
␣0␣0␣0␣0␣0␣0␣1␣0
LumOrb
Thrshld
0.5d-8␣0.5d-4␣1.0d-4
Iterations
␣50␣25
End␣of␣input
!ln␣-s␣$Home/C6H6.Ano␣␣␣␣␣ANO
!ln␣-s␣$Home/C6H6.RasOrb␣␣NAT001
!ln␣-s␣C6H6.OneInt␣␣␣␣␣␣␣␣ONE001
␣&GENANO␣&END
Title
␣Rydberg␣basis␣set␣for␣benzene.
sets
␣1
Center
X
Weights
␣1.0
end␣of␣input
```

Here is the shell script used for this example. It is written in Korn shell, but no exotic features of Korn shell are used, so rewriting them into C shell, or whatever your favorite shell is, is a straightforward matter.

```
#!/bin/ksh
Project='C6H6'
Home=$PWD
WorkDir=/temp1/$LOGNAME/$Project
export␣Project␣WorkDir
print␣'Start␣of␣job:'␣$Project
print␣'Current␣directory:'␣$Home
print␣'Scratch␣directory:'␣$WorkDir
```

```
#
trap␣'exit'␣ERR
rm␣-fr␣$WorkDir
mkdir␣␣$WorkDir
cd␣␣␣␣␣$WorkDir
#
molcas␣␣$Home/$Project.input␣>$Project.output
#
cd␣-
rm␣-r␣$WorkDir
```

For thiophene one can proceed in the same way. The only difference (apart from the fact that we generate s, p, d functions) is that two states of the cation are going to be computed and therefore the final step using the GENANO program will involve two files and have the following input:

```
!ln␣-s␣$Home/Thiophene.Ano␣␣␣␣␣ANO
!ln␣-s␣$Home/Thiophene.RasOrb1␣␣NAT001
!ln␣-s␣$Home/Thiophene.RasOrb2␣␣NAT002
!ln␣-s␣Thiophene.OneInt␣␣␣␣␣␣␣␣ONE001
!ln␣-s␣Thiophene.OneInt␣␣␣␣␣␣␣␣ONE002
␣&GENANO␣&END
Title
␣Rydberg␣basis␣set␣for␣thiophene.
sets
␣2
Center
X
Weights
␣0.5␣0.5
End␣of␣input
```

The charge centroid is chosen as an average of the charge centroids of the two cations.

SEWARD and CASSCF calculations.

Once we have built the diffuse basis set we can proceed with the SEWARD and CASSCF calculations of the different states. Remember that no quantitative result can be expected for calculations which use less than a DZP basis set. Additionally, as we are using methods which include large amounts of correlation, it is also recommended to use basis sets designed to include the correlation, such as the Dunning correlation-consistent basis sets or the Atomic Natural Orbital-type basis sets. Several tests of the accuracy of the ANO-type basis sets for calculations on excited states can be found elsewhere [48]. It was found that the minimum basis set suitable for calculations on excited states is the ANO $3s2p1d$ basis set for the first row atoms, with $2s$ functions for the hydrogen. The recommended basis however is an ANO $4s3p1d$ basis set.

We proceed with the calculations on thiophene. The inputs for the programs `SEWARD`, `SCF`, and `RASSCF` (^{1}A$_1$ states) are:

```
␣&SEWARD␣&END
Title
Thiophene␣molecule.␣Experimental␣gas-phase␣geometry.
Symmetry
␣X␣Y
Basis␣set
S.ANO-L...5s4p2d.
```

```
S1␣␣␣␣0.000000␣␣0.000000␣␣0.000000␣␣Bohr
End␣of␣basis
Basis␣set
C.ANO-L...4s3p1d.
C1␣␣␣␣0.000000␣␣2.333062␣␣2.246725␣␣Bohr
C2␣␣␣␣0.000000␣␣1.344416␣␣4.639431␣␣Bohr
End␣of␣basis
Basis␣set
H.ANO-L...2s1p.
H1␣␣␣␣0.000000␣␣4.288992␣␣1.677364␣␣Bohr
H2␣␣␣␣0.000000␣␣2.494694␣␣6.327573␣␣Bohr
End␣of␣basis
Basis␣set
X....1s1p1d␣/␣Inline
␣0.0000000␣2
*␣␣s-type␣diffuse␣functions
␣␣␣␣8␣␣␣␣1
␣.024624␣.011253␣.005858␣.003346␣.002048␣.001324␣.000893␣.000624
␣␣.38826283
-1.91720062
␣1.70115553
-2.69265935
␣3.15654806
-2.69329518
␣1.44320084
␣-.35712479
*␣␣p-type␣diffuse␣functions
␣␣␣␣8␣␣␣␣1
␣.042335␣.019254␣.009988␣.005689␣.003476␣.002242␣.001511␣.001055
␣␣.14713386
␣-.64370136
␣-.17112583
␣-.62433766
␣␣.58193247
␣-.53426167
␣␣.30777301
␣-.08250038
*␣␣d-type␣diffuse␣functions
␣␣␣␣8␣␣␣␣1
␣.060540␣.027446␣.014204␣.008077␣.004927␣.003175␣.002137␣.001491
␣␣.24501363
␣␣.04635428
␣␣.66592833
␣-.08963981
␣␣.52211247
␣-.32807746
␣␣.18219220
␣-.04616325
X␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣␣.0000000000␣␣␣␣␣␣␣␣␣␣.1609268500
End␣of␣Basis
End␣of␣Input


␣&SCF␣&END
Title
␣Thiophene␣molecule
Occupied
11␣1␣7␣3
Iterations
40
End␣of␣Input


!ln␣-fs␣$TempDir/$Project.1A1.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣pipi␣␣1A1␣states
```

```
Symmetry
␣␣␣␣1
Spin
␣␣␣␣1
Nactel
␣␣␣␣8␣␣␣␣0␣␣␣␣0
Frozen
␣␣␣␣4␣␣␣␣1␣␣␣␣3␣␣␣␣0
Inactive
␣␣␣␣6␣␣␣␣0␣␣␣␣4␣␣␣␣0
Ras2
␣␣␣␣1␣␣␣␣5␣␣␣␣0␣␣␣␣3
CiRoot
6␣6
1␣2␣3␣4␣5␣6
1␣1␣1␣1␣1␣1
Iter
50,25
LumOrb
End␣of␣Input
```

The wave function and natural occupation numbers obtained for the ^{1}A$_1$ states are:

```
                                  Wave function printout:
occupation of active orbitals, and spin coupling of open shells (u,d: Spin up or down)

printout of CI-coefficients larger than  0.38 for root  1
      energy= -551.412548
conf/sym 1 22222 444 Coeff  Weight
            11 2 22000 200   0.95720 0.91624

printout of CI-coefficients larger than  0.38 for root  2
energy= -551.192455
conf/sym 1 22222 444 Coeff  Weight
            14 2 22000 u0d   0.38522 0.14839
            20 2 2ud00 200   0.68777 0.47302

printout of CI-coefficients larger than  0.38 for root  3
energy= -551.178212
conf/sym 1 22222 444 Coeff  Weight
            85 2 2u0d0 200   0.74016 0.54783
    86 2 2u00d 200   0.46282 0.21421

printout of CI-coefficients larger than  0.38 for root  4
energy= -551.155996
conf/sym 1 22222 444 Coeff  Weight
            12 2 22000 ud0   0.49009 0.24019
            14 2 22000 u0d   0.72977 0.53257

printout of CI-coefficients larger than  0.38 for root  5
energy= -551.151801
conf/sym 1 22222 444 Coeff  Weight
    85 2 2u0d0 200  -0.48463 0.23486
    86 2 2u00d 200   0.78218 0.61180

printout of CI-coefficients larger than  0.38 for root  6
energy= -551.106218
conf/sym 1 22222 444 Coeff  Weight
     1 2 22200 000  -0.50027 0.25027
    20 2 2ud00 200  -0.49511 0.24514
    29 2 u2d00 200   0.46904 0.22000

Natural orbitals and occupation numbers for root 1
sym 1:   1.999604
sym 2:   1.991918  1.943992  0.097398  0.000219  0.000640
sym 4:   1.904095  0.061524  0.000611
```

```
Natural orbitals and occupation numbers for root 2
sym 1:   1.999436
sym 2:   1.947529  1.248261  0.788864  0.028171  0.000731
sym 4:   1.617765  0.032985  0.336259
Natural orbitals and occupation numbers for root 3
sym 1:   1.999273
sym 2:   1.926567  1.085938  0.128802  0.904415  0.000774
sym 4:   1.805386  0.141116  0.007730
Natural orbitals and occupation numbers for root 4
sym 1:   1.999591
sym 2:   1.938931  1.828828  0.185815  0.001667  0.027931
sym 4:   1.100050  0.074750  0.842438
Natural orbitals and occupation numbers for root 5
sym 1:   1.999251
sym 2:   1.935074  1.086440  0.103317  0.001139  0.911640
sym 4:   1.854839  0.074961  0.033340
Natural orbitals and occupation numbers for root 6
sym 1:   1.999766
sym 2:   1.874358  1.484874  1.099307  0.004906  0.008790
sym 4:   1.285113  0.235809  0.007076
```

We have only included the configurations with weights larger than 10%. Root one corresponds to the closed-shell ground state. To understand the character of the states one must also analyze the orbitals, remembering that the active orbitals are not ordered within the active space.

The following output shows the coefficients of the diffuse functions (center X) which appear in the *MOLCAS* output. Active orbitals two, three, and six in symmetry 2 are valence orbitals (they have main contributions from the other functions not printed here) and orbitals four and five are Rydberg orbitals. It is usual that they appear as mixed orbitals ($3p$-$3d$ here) but this mixing has no consequences on the excitation energies. This is also the reason why the Rydberg states appear not as clearly singly configurational states but mixed as in root 5 (see above).

```
   Molecular orbitals for symmetry species 2

   ORBITAL        2        3        4        5        6
   ENERGY      .0000    .0000    .0000    .0000    .0000
   OCC. NO.   1.8923   1.4570    .4122    .1674    .1689

19 X  2px     -.0203    .0055   -.0082    .8091    .4535
20 X  3d1+     .0064   -.0037   .0369    .4430  -1.0132

   Molecular orbitals for symmetry species 4

   ORBITAL        1        2        3
   ENERGY      .0000    .0000    .0000
   OCC. NO.   1.5865    .1722    .1439

15 X  3d2-     .0032    .5171    .9600
```

Both by looking at the configurations and the occupation numbers we can identify the states. Root two has a main configuration described by an excitation $3\mathrm{b}_1 \rightarrow 4\mathrm{b}_1$. As $4\mathrm{b}_1$ is a valence orbital, the resulting state will also be a valence state. Root three, on the contrary, has a main configuration $3\mathrm{b}_1 \rightarrow 5\mathrm{b}_1$, and $5\mathrm{b}_1$ is a Rydberg orbital. $3\mathrm{b}_1$ is the HOMO-1 orbital, therefore we can expect the state represented by root three to be the HOMO-1→$3p_x$ Rydberg state. So, why does configuration $3\mathrm{b}_1 \rightarrow 5\mathrm{b}_1$ contribute 21% to this wave function if a Rydberg state is just a singly excited state?. The answer is in the composition of the orbitals. Orbitals four and five are a mixture of p_x and d_{xz}, and the configurational description must reflect that.

In summary we can make a initial classification of the states:

Root 1: Ground state
Root 2: Valence $\pi \rightarrow \pi^*$ state
Root 3: Rydberg 3b$_1$→3p_x state
Root 4: Rydberg 3a$_2$→3d_{xy} state
Root 5: Rydberg 3b$_1$→3d_{xz} state
Root 6: Valence $\pi \rightarrow \pi^*$ state

Orbital two of symmetry 4 also deserves attention. It has large contributions from the diffuse functions, although the remaining non-printed coefficients are even larger. It is an orbital of mixed valence-Rydberg character. This can affect the description of the valence states. In the present system the problem is minor because the orbital does not strongly participate in the description of the valence states as it is shown by the configurations and the occupation numbers, but in other systems the effect is going to be larger as we shall show later.

One important difference between valence and Rydberg states is the diffuse character of the latter. We can analyze the orbital extension of the states. Valence states have an orbital extension (second Cartesian moment) similar to the ground state extension. Rydberg states, on the contrary, should have a diffuse character. Additionally we can also study the Mulliken population analysis. Both appear in the RASSCF output.

```
      Mulliken population Analysis for root number: 1

      Gross atomic populations per centre and basis function type

               S1      C1      C2      H1      H2      X
      Total 15.8153 12.3470 12.2660  1.6887  1.8021   .0809

      Expectation values of various properties for root number: 1

  2-nd Cartesian moments: origin at (   .00000000,   .00000000,  2.15947162)
 ----------------------------------------------------------------------------
 Component                            XX              YY              ZZ
 Total                      -30.24626427    -21.54920631    -24.73702724

      Mulliken population Analysis for root number: 2

      Gross atomic populations per centre and basis function type

               S1      C1      C2      H1      H2      X
      Total 15.6548 12.3730 12.1962  1.6914  1.8015   .2831
      Expectation values of various properties for root number: 2

  2-nd cartesian moments: origin at (   .00000000,   .00000000,  2.15947162)
 ----------------------------------------------------------------------------
 Component                            XX              YY              ZZ
 Total                      -42.75835009    -28.13902538    -28.72863222

      Mulliken population Analysis for root number: 4

      Gross atomic populations per centre and basis function type

               S1      C1      C2      H1      H2      X
      3d2-    .0334   .0306   .0413   .0000   .0000   .9662
      Total 15.5924 11.8522 12.0083  1.6814  1.7986  1.0671
      Expectation values of various properties for root number: 4

  2-nd cartesian moments: origin at (   .00000000,   .00000000,  2.15947162)
```

```
----------------------------------------------------------------------------
Component                            XX             YY              ZZ
Total                      -89.85913318   -76.33249740    -44.45493589

     Mulliken population Analysis for root number: 6

     Gross atomic populations per centre and basis function type

              S1      C1      C2      H1      H2      X
     Total 15.6154 12.4779 12.3182  1.6946  1.8028   .0911

     Expectation values of various properties for root number: 6

 2-nd cartesian moments: origin at (    .00000000,    .00000000,  2.15947162)
----------------------------------------------------------------------------
Component                            XX             YY              ZZ
Total                      -31.85163136   -24.13169375    -26.69322385
```

The Mulliken analysis provides us with the charge distribution per atom and basis function. If we have used for the Rydberg states singly centered Rydberg functions we can observe a population close to one on the X center. This is what happened in root four (see above). In addition we can see that the electron is placed in the 3d2- ($3d_{xy}$) Rydberg orbital, confirming the character of the state. The orbital extension is undoubtedly much larger in the fourth root than in the ground state. The second and sixth roots however have a much more compact description, especially the sixth, and they have low populations on center X. The second root is somewhat more diffuse but it can be still considered a clear valence state with minor Rydberg mixing.

It is very important to ensure that the relevant states of the symmetry are included in the CASSCF calculation. This may mean performing different experiments by increasing the number of roots and analyzing the results. Valence states are specially sensitive to this because they are high roots at the CASSCF level. Take for instance the sixth root. At the CASSCF level, it is 1.35 eV higher in energy than its preceding root. It could happen that other close Rydberg states or even valence states (such as mainly doubly excited states) were lower at this level of calculation. It can be also helpful to analyze the transition moment to be sure that the intense valence states are present in the set of computed states.

The RASSCF inputs for the remaining states replace the following keywords:

```
!ln␣-fs␣$TempDir/$Project.1B2.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣pipi␣␣1B2␣states
Symmetry
␣␣␣␣3
CiRoot
5␣5
1␣2␣3␣4␣5
1␣1␣1␣1␣1


!ln␣-fs␣$TempDir/$Project.1B1n.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣npi␣␣1B1␣states
Symmetry
␣␣␣␣2
CiRoot
1␣1
1
```

```
!ln␣-fs␣$TempDir/$Project.1A2n.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣npi␣␣1A2␣states
Symmetry
␣␣␣␣4
CiRoot
2␣2
1␣2
1␣1


!ln␣-fs␣$TempDir/$Project.1B1.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣pisigma␣␣1B1␣states
Symmetry
␣␣␣␣2
Ras2
␣␣␣␣5␣␣␣␣3␣␣␣␣2␣␣␣␣2
CiRoot
6␣6
1␣2␣3␣4␣5␣6
1␣1␣1␣1␣1␣1


!ln␣-fs␣$TempDir/$Project.1A2.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣pisigma␣␣1A2␣states
Symmetry
␣␣␣␣4
Ras2
␣␣␣␣5␣␣␣␣3␣␣␣␣2␣␣␣␣2
CiRoot
6␣6
1␣2␣3␣4␣5␣6
1␣1␣1␣1␣1␣1


!ln␣-fs␣$TempDir/$Project.1A1n.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣nsigma␣␣1A1␣states
Symmetry
␣␣␣␣1
Ras2
␣␣␣␣5␣␣␣␣3␣␣␣␣2␣␣␣␣2
CiRoot
4␣4
1␣2␣3␣4
1␣1␣1␣1
```

and use the corresponding links to save a *JOBIPH* file from each calculation.

We must ensure that the right orbitals are included into the active space. For instance, computing the $^1\mathrm{A}_2$ and $^1\mathrm{B}_1$ Rydberg states with the active space (5322) we observe that one Rydberg orbital is absent from the active space in both cases. For the $^1\mathrm{A}_2$ state it was orbital $3d_{yz}$. Instead, an extra-valence σ^* orbital took its place and therefore the sixth root of symmetry $^1\mathrm{A}_2$ was not the expected $2\mathrm{b}_1 \rightarrow 3d_{yz}$ Rydberg state. In this case we can reorder the orbitals including the Rydberg state in the active space and excluding the other orbital and make the calculation again. Hopefully the new calculation will include the Rydberg state into the selected roots. If not we can always increase the number of roots or increase the active space to have both orbitals included.

It is very important to remember that to compute energy differences one must always use states computed using the same active space. Therefore, if we are computing vertical excitation energies we must have the ground state energy computed in all the different active spaces employed. One can make the comparison using a ground state computed in the average procedure or as a single root. They do not differ significantly. For consistency, we will use a ground state computed as a single root. Therefore we have to perform two CASSCF calculations using the inputs where we replace:

```
!ln␣-fs␣$TempDir/$Project.11A1.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣Ground␣state␣(1503)
Symmetry
␣␣␣␣1
Ras2
␣␣␣␣1␣␣␣␣5␣␣␣␣0␣␣␣␣3
CiRoot
1␣1
1
```

```
!ln␣-fs␣$TempDir/$Project.11A1r.JobIph␣JOBIPH
␣&RASSCF␣&END
Title
␣Thiophene.␣Ground␣state␣(5322)
Symmetry
␣␣␣␣1
Ras2
␣␣␣␣5␣␣␣␣3␣␣␣␣2␣␣␣␣2
CiRoot
1␣1
1
```

CASPT2 calculations.

Once the reference wave functions have been computed at the CASSCF level we can perform the CASPT2 calculations. The *JOBIPH* file from each CASSCF calculation contains data that describes the state(s). If several CASSCF states are present on a *JOBIPH* file, then any of this may act as root function for the CASPT2. The input to the CASPT2 must then tell which one of the states we want. In previous *MOLCAS* version the keyword LROOT was used. Although it will still work, it has been substituted by the more convenient keyword MULTISTATE, which allows now to perform Multi-State CASPT2 calculations. We will start by discussing single state CASPT2 calculations:

```
␣&CASPT2␣&END
Title
␣caspt2␣input
MultiState
1␣1
End␣of␣input
```

The CASPT2 calculation will be performed on the ground state with the active space (1305), stored on the *JOBIPH* file that we named *$Project.11A1.JobIph*. The final full CASPT2 result is:

```
      Reference energy:        -551.4423376617
      E2 (Non-variational):       -.6341237973
```

```
      E2 (Variational):          -.6341237319
      Total energy:            -552.0764613935
      Residual norm:               .0000008080
      Reference weight:            .80657
```

For a perfectly converged result, the two formulae used to compute E2 are equivalent, but if there are (as is usually the case) a small residual error in the CASPT2 equation system, then the variational result is much more accurate. In particular, for numerical differentiation the variational energy should always be used. If a level shift has been used, in order to avoid singularities (see below), then the non-variational energy and the variational one will differ. The former is the conventional E2 as obtained with the modified (shifted) $\hat{H}_0$ operator, while the latter is a corrected value very close to what would have been obtained with the unshifted operator if the near-singular term had been removed. The latter energy is the one that should normally be used.

For the ground state with a reasonable active space, all coefficients in the first order wave function and all contributions to the second-order energy will be small. For excited states, large contributions may occur, and then the second-order perturbation treatment may be invalid. One criterion for a good calculation is that the reference weight should be close to that of the ground state. When this is not true, special remedies may be considered. For example, we compute the CASPT2 correction for the sixth root of symmetry one, using the *`JOBIPH`* file called $Project.1A1.JobIph. The input is:

```
␣&CASPT2␣&END
Title
␣caspt2␣input
MultiState
1␣6
End␣of␣input
```

and the result (always full CASPT2 results):

```
      Reference energy:        -551.1062184006
      E2 (Non-variational):      -.7460718503
      E2 (Variational):          -.7460719607
      Total energy:            -551.8520232128
      Residual norm:               .0000009146
      Reference weight:            .29470
```

We observe a low weight of 0.295 for the CASSCF reference, compared to the value 0.807 in the ground state. The low weight for the excited state is a warning sign: the second order treatment may be invalid. However, if so, the problem is due to one or a few specific terms in the first-order wave function.

In the output, there is a section with warnings for large contributions to the energy, low denominator values, or large coefficients.

```
CASE  SYM   ACT IND   NON-ACT INDICES  DENOMINATOR  RHS value  COEFFICIENT CONTRIBUTION

ATVX   2   Mu2.0001   Se2.007            .01778941  -.00706261   .72136097  -.00509469
ATVX   2   Mu2.0001   Se2.009            .20859986   .03118841  -.14372642  -.00448260
ATVX   4   Mu4.0001   Se4.004            .02156184  -.01357269  1.20409651  -.01634282
AIVX   1   Mu1.0001   In1.010 Se1.014    .08105563   .00023689  -.00197645  -.00000047
AIVX   1   Mu1.0001   In3.007 Se3.012    .28275882  -.02231776   .08282960  -.00184857
```

In CASPT2, the wave operator is a sum of two-electron excitations, $\sum C_{pqrs}\hat{E}_{pqrs}$, where the singlet excitation operator $\hat{E}_{pqrs}$ is normal-ordered and summed over spin. The electrons are transferred from s to r and from q to p.

No one-electron excitations are used. This is not due to any approximation; it is simply because, for a RASSCF root function with active electrons, the single excitations are exact linear combinations of the double excitations.

The non-orthogonality, as well as the non-diagonal terms of the $\hat{H}_0$, makes it difficult (and to some extent irrelevant) to obtain a label that partitions the wave function and correlation energy in terms of orbital indices of elementary excitations. However, the CASPT2 program uses internally an orbital system that diagonalizes part of the Fock matrix: the block diagonal part which does not include coupling between inactive, active and virtual orbitals. The first-order wave function, or equivalently the first-order wave operator, can be subdivided into terms that are grouped into eight different cases. These are named by four-letter combinations as follows. The letters A, B,C or D are used for secondary (virtual) orbitals; T, U, V, or X for active ones, and I, J, K or L for inactive orbitals. A case such as ATVX contains wave operator terms that can be written as $\hat{E}_{atvx}$, where a is a virtual orbital and t, v, and x are active.

The first-order wave function can be subdivided into individual terms labeled by the case (e.g. ATVX), the individual non-active orbital indices, and an active superindex that labels a linear combination of terms with different active orbital indices. The linear combination will 'mix' all active indices or index combinations within the case (with symmetry restrictions, if any) in such a way that *the individual terms that are used internally in the CASPT2 programs are orthogonal, and they diagonalize the block-diagonal part of* $\hat{H}_0$.

Of course, the complete $\hat{H}_0$ is used to solve the CASPT2 equations, which is why an iterative procedure is needed. However, in the diagnostic output above, the "DENOMINATOR" value is that of the resolvent of the block-diagonal part of $\hat{H}_0$. However, for diagnostics, this is a good approximation. (That it is not exact only shows by the fact that singularities in the energy do not occur exactly when the "DENOMINATOR" reported is equal to 0.)

The orbitals are labeled by the symmetry type, a period, and then the ordering number within that symmetry type. However, for clarity, it also is prefixed by the letters "Fr", "In", "Ac", "Se" or "De" for frozen (uncorrelated), inactive, active, secondary, and deleted orbitals. In the wave operator, the only possible orbital labels are "In" and "Se". The active superindex is given in formulae as μ, ν, etc so it is given a prefix "Mu".

Most of the cases are further subdivided into a plus and a minus linear combination making altogether 13 cases. Thus, the BVAT case is subdivided into BVATP and BVATM, containing terms of the type $\hat{E}_{bvat} \pm \hat{E}_{avbt}$, respectively. This has nothing to do with spin. It offers some technical advantages in the equation solution.

For more details see Refs. [49, 50, 51]

The first configuration shown in the thiophene output involves the excitation from the active space to the secondary orbital, which is orbital nr seven of symmetry two (Se2.007). The denominator value for this configuration is close to zero (0.01778941). This is an energy difference, in the $\hat{H}_0$ approximation. Thus the root state, and some eigenstate of $\hat{H}_0$ in the interacting space, have almost the same energy value.

Table 5.13: Labeling for the configurations in caspt2.

Config.	Excitation 1	Excitation 2
VJTU	Inactive (J) → Active (V)	Active (U) → Active (T)
VJTIP	Inactive (J) → Active (V)	Inactive (I) → Active (T)
VJTIM	Inactive (J) → Active (V)	Inactive (I) → Active (T)
ATVX	Active (T) → Secondary (A)	Active (X) → Active (V)
AIVX	Inactive (I) → Secondary (A)	Active (X) → Active (V)
or:	Active (X) → Secondary (A)	Inactive (I) → Active (V)
VJAIP	Inactive (J) → Active (V)	Inactive (I) → Secondary (A)
VJAIM	Inactive (J) → Active (V)	Inactive (I) → Secondary (A)
BVATP	Active (V) → Secondary (B)	Active (T) → Secondary (A)
BVATM	Active (V) → Secondary (B)	Active (T) → Secondary (A)
BJATP	Inactive (J) → Secondary (B)	Active (T) → Secondary (A)
BJATM	Inactive (J) → Secondary (B)	Active (T) → Secondary (A)
BJAIP	Inactive (J) → Secondary (B)	Inactive (I) → Secondary (A)
BJAIM	Inactive (J) → Secondary (B)	Inactive (I) → Secondary (A)

Such states, that were not included in the CASSCF configuration interaction but have energies within the range of the lowest CAS states, cause frequent problems in excited state calculations, since they often give small denominators and even, at particular geometries, singularities. We call these states intruders, by analogy to a similar phenomenon in multistate perturbation theory. A calculation of excited states by means of a perturbation theory based on an active space has to deal with the problem of intruder states. This is especially common when large and diffuse basis sets, such as the Rydberg functions, are included in the calculations.

In this example, the coefficient to the first order wave function is large (0.72136094). So is the contribution to the second order energy (-0.00509469 H), -0.14 eV. Even worse is the situation for the third term printed involving the fourth orbital (secondary) of symmetry four with an energy contribution of 0.44 eV. The analysis of the secondary orbitals $7b_1$ and $4a_2$ (they are the first virtual orbital of their symmetry) indicates that they are extremely diffuse orbitals with large Rydberg character. Remember that the subspaces we are using are: frozen (4130), inactive (6040), and active (1503).

This is not the case in the other configurations shown. First we have other ATVX terms including the excitation to the secondary orbital Se2.009. Also we have an AIVX term, involving the excitation from inactive In3.007 to secondary Se3.012. Their contributions to the second order energy, -0.00448260 and -0.00184857, respectively, are not caused by accidental near degeneracies in the value of the denominator. The orbitals involved are not of Rydberg character either. We have finally included as an example the excitation AIVX involving the excitation from In1.010 to Se1.014. Although it has a small value for the denominator, its contribution to the second order energy is very small and therefore it does not represent an important problem.

Intruders can be eliminated by including sufficiently many orbitals in the active space. When this is a reasonable alternative, it is the preferred solution. Limitations in the number of active orbitals can make this approach impractical. However, especially when intruders have clear Rydberg character, their effect on the second-order energy is often small, except perhaps in a small range of geometries around a singularity due to accidental degeneracy. In this common situation, two other remedies are available: shifting the $\hat{H}_0$ Hamiltonian, or deleting virtual orbitals. These remedies will be described in some detail in the following.

In order to obtain continuous potential energy functions, one cannot use a case-by-case approach, such as deleting an orbital. However, the $\hat{H}_0$ can be modified in such a way as to eliminate weak singularities. A well-tested method is a level-shift technique called LS-CASPT2[43, 52]. A constant parameter is added to the external part of the zeroth-order Hamiltonian. Any denominator close to zero is thus shifted away from zero, and does not produce any singular term. Of course, in a worst-case scenario, it might happen that some other denominator, previously non-zero, is shifted to come close to zero. In general, it is the higher excited states, in combination with large diffuse basis sets and exploration of a large range of geometries, that is the greatest risk for troublesome intruders.

There is also a new, less tried technique, called the imaginary shift method [53]. Here, the use of an imaginary shift value (but taking the real part of the computed correlation energy) offers some advantage, since an imaginary shift cannot introduce new singularities.

With either of the level shift methods, the (2nd order) correlation energy E_2 and the (1st order) wave function will depend on the level shift used. A correction of therefore applied, whereby in practice this dependence is made small, except of course for the spurious term that has disappeared. The corrected energy is in fact computed by using Hylleraas' 2nd-order variational formula to evaluate E_2, with the *unshifted* $\hat{H}_0$,

$$E_2 = 2\langle\Psi_1|\hat{H}|\Psi_0\rangle + \langle\Psi_1|\hat{H}_0|\Psi_1\rangle \tag{5.3}$$

which we call the *variational* E_2 in the output listing.

To minimize the effect on relative energies, we recommend that the same level shift is used for all states and geometries, if possible. This may require some experimenting. A criterion on absence of disturbing intruders is that the weight of the reference wave function should be roughly the same in all calculations. Without shift, a difference of up to 10% between the weights of the ground and an excited state can be acceptable (that is, the excitation energy is accurate enough) in a `CASPT2` calculation without level shift. Using level shift, this should be adjusted to find a better match of reference weights. A detailed explanation of how to use the level-shift technique has been published [54]. Here we will simply summarize the main aspects.

Using the same *`JOBIPH`* file as before we perform a new CASPT2 calculation using the input:

```
␣&CASPT2␣&END
Title
␣caspt2␣input
MultiState
1␣6
Shift
0.1
End␣of␣input
```

A level-shift of 0.1 Hartree has been introduced as a separation of the eigenvalues of the zeroth-order Hamiltonian. The final energy is then corrected, and the result is:

```
Reference energy:        -551.1062184006
E2 (Non-variational):       -.6921992859
Shift correction:           -.0334372801
```

```
      E2 (Variational):          -.7256365659
      Total energy:            -551.8315878181
      Residual norm:               .0000003986
      Reference weight:            .74942

 CASE  SYM   ACT IND   NON-ACT INDICES  DENOMINATOR  RHS value  COEFFICIENT CONTRIBUTION

 ATVX   2   Mu2.0001   Se2.007            .01778941  -.00706261   .06072347  -.00042887
 ATVX   2   Mu2.0001   Se2.009            .20859986   .03118841  -.09700134  -.00302532
 ATVX   4   Mu4.0001   Se4.004            .02156184  -.01357269   .11838970  -.00160687
 AIVX   1   Mu1.0001   In3.007 Se3.012    .28275882  -.02231776   .05918658  -.00132091
```

Several details come to our attention. Firstly, the final CASPT2 energy is higher than the result with level-shift 0.0. This is because the introduction of the parameter decreases the amount of dynamical correlation included. Secondly, the weight of the reference function has increased greatly, from 0.29 to 0.74, meaning that the most important intruder states have been removed from the treatment. Finally, we can observe the new contributions of the printed configurations to the second order energy. Configurations involving excitations to the $7b_1$ and $4a_2$ orbitals have drastically decreased their contributions, proving that the previous contributions were due to degeneracies in the denominators. However, the other two configurations remain almost as they were before, only slightly decreasing their contributions.

Now we use a value for the level-shift parameter of 0.2 Hartree:

```
      Reference energy:        -551.1062184006
      E2 (Non-variational):      -.6619040669
      Shift correction:          -.0557159229
      E2 (Variational):          -.7176199898
      Total energy:            -551.8235712419
      Residual norm:               .0000009298
      Reference weight:            .78212

 CASE  SYM   ACT IND   NON-ACT INDICES  DENOMINATOR  RHS value  COEFFICIENT CONTRIBUTION

 ATVX   2   Mu2.0001   Se2.007            .01778941  -.00706261   .03193515  -.00022555
 ATVX   2   Mu2.0001   Se2.009            .20859986   .03118841  -.07304944  -.00227830
 ATVX   4   Mu4.0001   Se4.004            .02156184  -.01357269   .06238180  -.00084669
 AIVX   1   Mu1.0001   In3.007 Se3.012    .28275882  -.02231776   .04673419  -.00104300
```

The observed tendencies are maintained. Finally, a value of 0.3 Hartree:

```
      Reference energy:          -551.1062184006
      E2 (Non-variational):        -.6347955450
      Shift correction:            -.0735679820
      E2 (Variational):            -.7083635270
      Total energy:              -551.8145819276
      Residual norm:                 .0000006328
      Reference weight:              .80307

 CASE  SYM   ACT IND   NON-ACT INDICES  DENOMINATOR  RHS value  COEFFICIENT CONTRIBUTION

 ATVX   2   Mu2.0001   Se2.007            .01778941  -.00706261   .02173413  -.00015350
 ATVX   2   Mu2.0001   Se2.009            .20859986   .03118841  -.05865340  -.00182931
 ATVX   4   Mu4.0001   Se4.004            .02156184  -.01357269   .04240583  -.00057556
 AIVX   1   Mu1.0001   In3.007 Se3.012    .28275882  -.02231776   .03862959  -.00086213
```

The contributions to the energy are much lower for each increase of the parameter, but we must never forget that we are loosing dynamical correlation with the increase of the level-shift factor. In a calculation of excitation energies that means that the resulting excitation energies become larger each time (dynamical correlation is larger in the excited state). Therefore, the level-shift parameter must be set to the lowest possible value which solves the intruder state problems. In practice it is then convenient to scan all the valence states for several values of the parameter and look for two factors:

- Reference weight as close as possible to the ground state reference weight with the same level shift parameter (LS).
- Excitation energies (ES) as stable as possible with the increment of the level-shift parameter (LS).

We now compute the ground state (GS) also for the level-shift values of 0.1, 0.2, and 0.3, and compare the excitation energies ΔE (always between states computed with the same parameter):

Table 5.14: Excitation energies and reference weights of thiophene for different level shift values.

LS (H)	ΔE (eV)	weight GS	weight ES
0.0	6.11	0.81	0.29
0.1	6.64	0.82	0.75
0.2	6.79	0.83	0.78
0.3	6.89	0.84	0.80

After checking the remaining states we conclude that a level shift of 0.1 Hartree is enough for our purposes. However the results seem to be too unstable with respect to the increase of the level-shift parameter. As our active space only comprises nine orbitals, we can consider the possibility of increasing it by including two more active orbitals in symmetries b_1 and a_2. In this way we minimize the intruder states problems in the best way, by introducing extra (not diffuse hopefully) orbitals. This will increase the accuracy.

The introduction of a (real) level-shift parameter does not automatically remove intruder state problems. It happens that a shift leads to more severe problems that those observed without level-shift. Examples and further explanations are given in e.g. ref. [54]. In such a case is may be possible to find a range of level-shift values where none of the computed states present intruder state problems. In a few cases we have found it necessary to use a shift larger than 0.3 Hartree. Another solution is to try an imaginary shift. This option has not been extensively investigated yet.

Consider a situation like the following:

```
CASE  SYM   ACT IND   NON-ACT INDICES  DENOMINATOR  RHS value  COEFFICIENT CONTRIBUTION

ATVX   2   Mu2.0001   Se2.004          -.30281661  -.00194108  -.37224517   .00072256
```

This is a calculation performed using level shift of 0.3 H. (The approximate denominator printed in the listing is that *without* the added shift). We have added the level shift to solve intruder states problem in other states, but we should use the same technique for all the computed states for consistency reasons (of course always using a ground state computed with the same level shift value). We find, however, that the weight of the CASSCF reference function is lower in the case with level shift 0.3 H (0.61) than in the case without level shift (0.69). In this state we have a denominator with a value close to -0.3 H. As the level shift we apply is a positive quantity (0.3 H) added to this denominator, we have created a problem by decreasing the denominator to a value close to zero. The coefficient of the configuration increases, which is reflected in the contributions to the second-order energy. Therefore, before applying any level shift, it is wise to check the values of the most important denominators to see if any of them is going to be close to the value of the applied level shift. In those situations we should set the level shift to another value. Sometimes the consequences for the final energy are small (here for instance) but this is not always the case (see ref. [54]).

It is also possible to delete virtual orbitals. This is occasionally used, e.g. when using other types of basis sets than ANO's, in order to delete virtual orbitals that are core-correlating. The procedure to do that is to take an orbital file, such as that produced by SCF or RASSCF, and edit it by hand and then using it as *`INPORB`* file in the RASSCF step. The orbitals one wants to delete are placed at the end of their symmetry group, and the keyword DELETED in used the RASSCF input, indicating how many orbitals are going to be deleted by symmetry. The program will ignore the deleted orbitals, both in RASSCF and the subsequent CASPT2 steps. To obtain accurate energy differences it is necessary to use the same set of initial orbitals and recompute the ground state (or the state one is comparing with) with the same number of deleted orbitals.

When the above scheme is used in order to try to eliminate intruders in CASPT2, the best way is if the *`INPORB`* can be prepared from the CASPT2 calculation where the intruder problem occurred.

For that calculation, the natural orbital analysis that follows the CASPT2 calculation shows up a virtual orbital with abnormally large occupation number and diffuse character. Use an editor to move this orbital to the end of the orbital file, and use it as *`INPORB`*. When the calculation is repeated, intruders with this orbital heavily populated have been eliminated. Occasionally, several orbitals need to be removed.

The deletion of virtual orbitals works best at single-geometry calculations, such as obtaining the vertical electronic spectrum.

Let us focus on the Multi-State CASPT2 type of calculations. The original reference [13] should be carefully read before using the method. This multidimensional perturbative approach considers the coupling of a number of CASPT2 states, a condition which is crucial to solve certain problems such as adiabatic crossing among states, strong valence-Rydberg situations, etc. The treatment is performed for a number of roots of the same symmetry provided they originate from a previous State-Average CASSCF calculation, that is, the `CASPT2` program will use the binary *`JOBIPH`* file from a previous SA-CASSCF calculation, for instance, the six roots $^1A_1$ CASSCF calculation in thiophene. The corresponding `CASPT2` input to treat simultaneously the six states will be:

```
␣&CASPT2␣&END
Title
␣mscaspt2␣input
MultiState
6␣1␣2␣3␣4␣5␣6
Shift
0.3
End␣of␣input
```

A level shift parameter of 0.3 au has been selected for comparison with the previous calculations. The program creates a new binary file, *`JOBMIX`*, which contains the newly generated Perturbatively Modified (PM) CASSCF wave function.

Using the previous input, the `CASPT2` module will perform in a single run six consecutive single-root CASPT2 calculations for each one of the CASSCF states. At the end of each of the calculations the contributions to the Hamiltonian coupling elements between the computed and the remaining states will be printed. After computing the six CASPT2 roots, the MS-CASPT2 treatment will be performed. First, the effective Hamiltonian matrix, asymmetric and symmetric, is printed.

```
Effective Hamiltonian matrix (Symmetric):
                 1  2         3 4         5
1    -.07013926
2    -.01263691     .12976380
3     .00071175     .01001560     .18051855
4     .00509735     .00990244    -.00321669     .19922802
5     .00607124     .00070650    -.00129815    -.00225583     .21601193
6     .01998132     .02350235    -.00771000    -.01037132    -.00264941
                 6
1     .18541807
```

Notice that the diagonal elements of the matrix correspond to the single root CASPT2 state energies, where some quantity, 551.0 au here, has been added to get a better print of the output. Following, the eigenvalues and eigenvectors of the diagonalized matrix are obtained:

```
Energies and eigenvectors:
  -552.07305076  -551.88140802  -551.81866833  -551.80756578  -551.79500203
      .99308520     -.10131857      .01038991      .05207094     -.02055799
      .07343489      .90295279      .31190606      .28061095     -.05245262
    -.00869768     -.19493901      .90626880     -.37241673      .03796203
    -.02478279     -.15572120      .13596794      .50373403      .83205915
    -.02204833     -.01553573      .05330075      .08679334      .05789830
    -.08492920     -.33454317      .24485766      .72011863     -.54745806
  -551.78350398
      .01655899
    -.02245882
    -.02155609
    -.10285444
      .99274682
    -.05129770
```

The eigenvalues correspond to the final MS-CASPT2 energies, while the eigenvectors describe the combination of the coupled CASPT2 state which give rise to the final MS-CASPT2 states. **Important:** Notice that the states are written in an increasing energy order, and therefore they do not, in general, correspond to the order obtained in the previous SA-CASSCF calculation. For instance, the MS-CASPT2 state number six, energy -551.78350398 au, mainly correspond to the fifth state of the previous calculation. It is very important to remember that the final states are linear combinations of the preceding ones, and therefore a one to one correspondence is hardly possible. In the present example most of the MS-CASPT2 states have a strong weight in just one of the preceding states, but this is not the case in many situations. Following in the output, a printing of the new wave function is obtained. It corresponds to linear combinations of the SA-CASSCF CI wave functions, obtained in the basis of the previous CASSCF averaged orbitals.

```
 The CI coefficients for the MIXED state nr.  1
------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
```

```
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
    11  2 22000 200 .960835       .923204
 The CI coefficients for the MIXED state nr.  2
----------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
    20  2 2ud00 200 .856751        .734023
 The CI coefficients for the MIXED state nr.  3
----------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
    85  2 2u0d0 200 .764848        .584993
    86  2 2u00d 200 .507350        .257404
 The CI coefficients for the MIXED state nr.  4
----------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
     1  2 22200 000   -.368003        .135427
    14  2 22000 u0d .732276        .536229
 The CI coefficients for the MIXED state nr.  5
----------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
     1  2 22200 000 .416925        .173826
    12  2 22000 ud0 .549793        .302272
    14  2 22000 u0d .455052        .207072
 The CI coefficients for the MIXED state nr.  6
----------------------------------------------------------------------------
CI COEFFICIENTS LARGER THAN 0.36
 Occupation of active orbitals, and spin coupling
 of open shells. (u,d: Spin up or down).
  Conf Occupation   Coef Weight
    85  2 2u0d0 200    -.517972       .268295
    86  2 2u00d 200     .776117       .602358
```

The comparison of the present wave functions, that will be hereafter called Perturbatively Modified (PM) CASSCF wave functions, and the previous CASSCF wave functions leads to several conclusions. Remember that the orbital basis has not changed, therefore those mixing related to the orbitals are not going to disappear. For instance, state number three will still be formed by two configurations, because the Rydberg 3px character is still delocalized between orbitals 5 and 6 or symmetry b_1. However the character of the second root has changed dramatically. Now one single configuration describes the state, which has acquired a very clear valence character. The previous mixing with a Rydberg-like configuration has disappeared. It is illustrative to carry out an additional analysis of the obtained states using the generated file *`JOBMIX`* as input file to perform a `RASSI` calculation, in which new PM-CASSCF properties for the states will be obtained. Even when the changes in energies are small, changes in the properties can be considerable. `RASSI` provides different types of matrix elements (see next section), and dipole moments, transition dipole moments and their directions, and orbital extensions (all of them available from the `RASSI` output) will be crucial for our purposes in the study of excited states.

Finally, it is necessary to remember that the extent of the MS interaction relies on the mixing of the previous states. This depends on different factors. The basis sets is one of them. The use of one or other atomic basis set to describe the diffuse functions may lead to different answers. It is not uncommon that CASPT2 results with different diffuse basis sets give different answers due to different extents of the valence-Rydberg mixing. It will be necessary to perform final MS-CASPT2 calculations. Those will change the CASPT2 result in some cases, but it will be unaffected in other cases. Another effect comes from the use of the level shift. The use of MS-CASPT2 does not prevent or affect the extent of the intruder effects. Remember that this effect is already included both in the diagonal terms of the effective Hamiltonian as in the non-diagonal coupling terms. Still a careful checking of different LS values and how they affect the CASPT2 values must be performed, and the final MS-CASPT2 results should be those in which the effect of the intruder states is small, always trying to use as low level shift values as possible. An alternative is to use an imaginary level shift. Finally, the extent of the off-diagonal coupling elements and its asymmetric character introduce further inaccuracies in the treatment. In most cases the proper enlargement of the active space diminishes most of the spurious effects and increases the accuracy.

Transition dipole moment calculations.

One powerful tool included in the *MOLCAS* package is the `RASSI` program. RASSI (RAS State Interaction) forms matrix elements of the Hamiltonian and other operators in a wave function basis which consists of individually optimized CI expansions from the RASSCF program. It also solves the Schrödinger equation within the space of these wave functions. In spectroscopy we need to compute the matrix elements of a one-electron operator such as the dipole transition moment to obtain the intensity of the transitions. In an absorption process this means computing the interaction of the ground state with the excited states. RASSI will compute all matrix elements among the states provided they have been computed with the number of inactive and active orbitals, and using the same basis set. The transition dipole moments are computed using the length representation.

In our example we have used two different active spaces. We therefore need to perform at least two RASSI calculations. First we will compute the interaction of the ground state 1^1A$_1$ (computed as single root), with the $\pi \rightarrow \pi^*$ ^{1}A$_1$ and ^{1}B$_2$ excited states. We should link the corresponding *`JOBIPH`* files:

```
ln -fs $Project.11A1.JobIph JOB001
ln -fs $Project.1A1.JobIph JOB002
ln -fs $Project.1B2.JobIph JOB003
```

and use the RASSI input file:

```
␣&RASSI␣&END
Nrofjobiphs
␣3␣1␣5␣5
␣␣1
␣␣2␣3␣4␣5␣6
␣␣1␣2␣3␣4␣5
End␣of␣input
```

As we are using states that are not orthogonal (this is the case among the 1^1A$_1$ ground state computed as a single root and the other ^{1}A$_1$ states) we must take the matrix elements of the transition dipole moment computed after the transformation to the eigenbasis; the second time they appear in the output:

```
  PROPERTY: MLTPL  1   COMPONENT:   2
  ORIGIN    :  .00000000D+00  .00000000D+00  .00000000D+00
  STATE     :       1             2             3             4

      1        .00000000D+00  .00000000D+00 -.43587844D+00  .00000000D+00
      2        .00000000D+00  .00000000D+00 -.10019699D+01  .00000000D+00
      3       -.43587844D+00 -.10019699D+01  .00000000D+00 -.46859879D+00
      4        .00000000D+00  .00000000D+00 -.46859879D+00  .00000000D+00
      5        .90773544D-01  .75718497D-01  .00000000D+00  .27645327D+00
      6        .00000000D+00  .00000000D+00  .41227462D+01  .00000000D+00
      7        .00000000D+00  .00000000D+00  .89741299D+00  .00000000D+00
      8       -.16935368D+00  .15487793D+01  .00000000D+00 -.41013917D+01
      9        .81381108D+00  .79559359D+00  .00000000D+00 -.88184724D-01
     10        .00000000D+00  .00000000D+00 -.43659784D+00  .00000000D+00
     11        .13520301D+01  .50454715D+00  .00000000D+00  .56986607D-01

 ...

  PROPERTY: MLTPL  1   COMPONENT:   3
  ORIGIN    :  .00000000D+00  .00000000D+00  .22419033D+01
  STATE     :       1             2             3             4
      1        .28126942D+00 -.92709234D+00  .00000000D+00  .11876829D+00
      2       -.92709234D+00  .26218513D+00  .00000000D+00  .14100968D+00
      3        .00000000D+00  .00000000D+00  .52558493D-01  .00000000D+00
      4        .11876829D+00  .14100968D+00  .00000000D+00  .36996295D+00
      5        .00000000D+00  .00000000D+00 -.43197968D+01  .00000000D+00
      6       -.15470487D+00 -.42660550D+00  .00000000D+00  .94593876D+00
      7       -.18676753D-01  .18738780D+01  .00000000D+00 -.37737952D+01
      8        .00000000D+00  .00000000D+00 -.28182178D+00  .00000000D+00
      9        .00000000D+00  .00000000D+00  .38253559D+00  .00000000D+00
     10        .12859613D+01  .48476356D+00  .00000000D+00  .35525361D+00
     11        .00000000D+00  .00000000D+00 -.39325294D-01  .00000000D+00
```

We have a symmetric matrix containing the results. The matrix elements corresponding to the interaction of the first state in the input (ground state) and the remaining states appear both in the first column and in the first row (only partially printed here). Remember that the transition dipole moment (TDM) matrix elements are determined by the symmetry. The matrix element $\langle {}^1\mathrm{A}_1|\mathrm{TDM}|{}^1\mathrm{A}_1\rangle$ will be zero for the x and y components of TDM, and non-zero otherwise. The matrix element $\langle {}^1\mathrm{A}_1|\mathrm{TDM}|{}^1\mathrm{B}_2\rangle$ will be non-zero only for the y component of TDM. This is because the product (wave function 1 × dipole moment component × wave function 2), if decomposed into irreducible representations, must contain the totally symmetric representation to have an allowed transition. In this simple case, we can use a multiplication table for the irreps. Thus, for instance, (${}^1\mathrm{A}_1$ (z) $\times$ TDM_y $\times$ ${}^1\mathrm{A}_1$ (z)) gives y, which does not belong to the totally symmetric representation. A look at the character table and the behavior of the x, y, z functions will give us the information we need.

Therefore, in the component two (y) of the transition dipole moment matrix elements we have zero values for the interaction among ${}^1\mathrm{A}_1$ states and non-zero values for the interaction among ${}^1\mathrm{A}_1$ and ${}^1\mathrm{B}_2$ states.

The RASSI program in 6.0 and later versions of *MOLCAS* will print the oscillator strengths and the Einstein A coefficients for all transitions. Also the angles of the transition moment vectors to the coordinate axes will be printed. In the calculation RASSI will use the energies given as input, so be careful to use the keywords HDIAG or EJOB to use energies which include dynamic correlation.

We illustrate how the oscillator strengths are computed. The 11 states are ordered by CASSCF energies. We focus on the valence states; firstly the fourth and fifth ${}^1\mathrm{B}_2$ states. Their transition dipole moment values in atomic units are 0.81381108 and 0.13520301D+01, respectively. The oscillator strength is defined as:

$$f = \frac{2}{3}(TDM)^2 \Delta E \tag{5.4}$$

The energy difference ΔE is the excitation energy expressed in atomic units. The transition moments were computed by CASSCF. It is usually not practically possible to compute them with dynamic correlation included, except if a common set of orbitals are used. However, the CASSCF values are usually good enough. (Exceptions occur, e.g. close to narrowly avoided crossings or conical intersections). The excitation energies, on the other hand, are quite sensitive to dynamic correlation. Thus, it is a good approach to use CASSCF TDMs and CASPT2 excitation energies. The values for the oscillator strengths of the two $^1B_2$ valence states are 0.086 and 0.324, respectively. The excitation energies are 5.31 and 7.23 eV, respectively. All data corresponds to results obtained using the 0.1 Hartree value for the level-shift parameter.

Remember that in other symmetries like C_{2h} the $^1B_2$ states have two components of TDM, x and y, for which the matrix elements with respect to the ground state are non-zero. In this case the TDM^2 value is computed as TDM$_x^2$ + TMD$_y^2$. In those cases is is also possible to compute the direction of the total TDM vector by taking their components and compute the angle respect to any of the axis.

You will find the complete calculation of the absorption spectrum of thiophene in reference [31]. You can observe that, despite there being no level-shift technique used, the final results on the excitation energies agree to within 0.1 eV to those shown here.

5.5.2 Influence of the Rydberg orbitals and states. One example: guanine.

Thiophene has a valence π, π^* orbital space small enough to allow the simultaneous inclusion of all the corresponding Rydberg orbitals into the active space (remember valence space (1302) + Rydberg spaces (0201) or (4020)), but this is not always the case. In addition, the valence-Rydberg mixing is not severe. This mixing is reflected in the orbital extension or the population analysis. In difficult cases valence and Rydberg orbitals mix, and then the configurations also mix. Valence states become more diffuse and Rydberg states more compact. Energetically this has minor consequences for the Rydberg states, which can be computed using these CASSCF mixed wave functions. This is not the case for the valence states. They are extremely sensitive to the mixing. Therefore, if we do not observe clear and compact valence states some mixing has occurred.

We consider the example of the guanine molecule, the nucleic acid base monomer. It is a system with 11 valence π, π^* orbitals which should be included into the active space. It is a planar system in the C_s point group. Focusing only in the $\pi \rightarrow \pi^*$ states we can label the active orbital space (0,11) where 0 is the number of a$'$ orbitals and 11 the number of a$''$ orbitals. In C_s symmetry the Rydberg orbitals are distributed as (6,3), using the same labeling. Therefore the calculation of the corresponding A$'$ states should use the space (0,14) with 14 active electrons and a large number of roots. This is a large calculation that one might want to avoid. One can perform several test calculations (maybe even RASSCF calculations) and find if any orbitals can be excluded. The lowest occupied π orbital is a deep orbital which does not participate in the lowest valence excited states and can be excluded from the active space. Despite this exclusion, a (0,13) orbitals calculation is still expensive.

We can proceed in another way. Consider the new valence space (0,10), and add only one more orbital designed to include the first Rydberg orbital. With this space of (0,11) orbitals and 12 active electrons we perform a CASSCF including 6 roots.

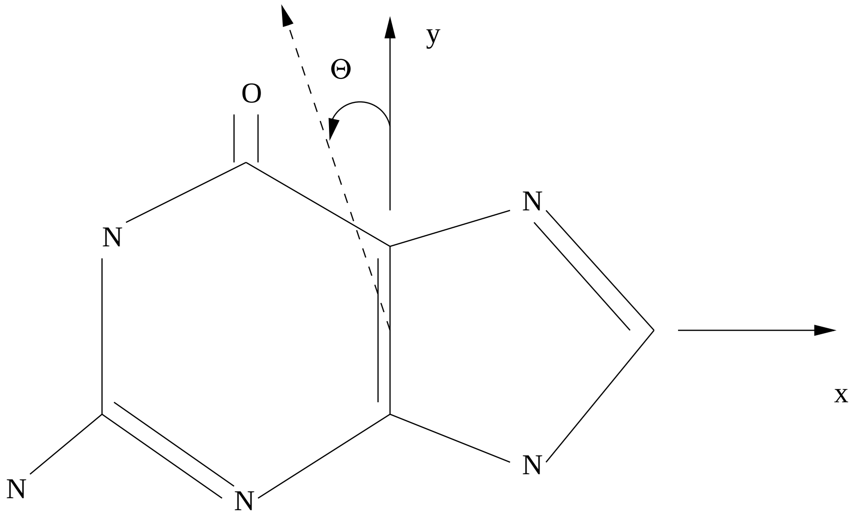


Figure 5.12: Guanine

Our basis set is of the ANO-L type contracted to C,N,O $4s3p1d$ / H $2s$, plus $1s1p1d$ optimized diffuse functions placed in the cation charge centroid. The results are collected in Table 5.15.

Table 5.15: CASSCF and CASPT2 excitation energies (eV), oscillator strengths (f), dipole moments (μ(D)), and transition moment directions (Θ) of singlet valence excited states of guanine[a]. The Rydberg orbitals have not been included in the active space.

State	Theoretical					Experiment[b]		
	CAS	*PT*2	*f*	Θ	μ	ΔE	*f*	Θ
π–π^* transitions								
2^1A$'$	5.72	4.47	.20	-64^o	1.07	4.4-4.5	.16	(-4^o,35^o)
3^1A$'$	6.74	5.30	.09	$+52^o$	2.72	4.9-5.0	.25	(-75^o)
4^1A$'$	7.18	5.63	.05	-90^o	3.10	5.7-5.8	<.05[c]	
5^1A$'$	8.45	6.83	.26	0^o	3.20	6.1-6.3	.41	(-71^o,-79^o)

[a] See ref. [55] for details.

There are important discrepancies between theoretical and experimental results, more important in the properties such as the intensities and the transition dipole moments than in the excitation energies. If we analyze the CASSCF output everything is apparently correct: six converged roots, all of them clear valence states, and no Rydberg orbital into the active space. This is the problem. At least one of the Rydberg orbitals should have been introduced into the active space. Rydberg and valence orbitals must be treated simultaneously and this is not possible if there is no Rydberg orbital in the active space.

The correct way to proceed is to take the first Rydberg orbital ($3p_z$) and place it as the 11th active orbital of a$''$ symmetry. Then the CASSCF calculation will retain it in the space. Once the calculation has converged we observe than at least one of the computed states is of Rydberg character. It can also happen that some mixing appears in the valence states

Table 5.16: CASSCF and CASPT2 excitation energies (eV), oscillator strengths (f), dipole moments (μ(D)), and transition moment directions (Θ) of singlet valence excited states of guanine[a,b]. The Rydberg orbitals have been first included in the active space and then deleted.

State	Theoretical					Experiment		
	CAS	*PT*2	*f*	Θ	μ	ΔE	*f*	Θ
π–π^* transitions								
$2^1A'$	6.08	4.76	.133	-15^o	7.72	4.4-4.5	.16	$(-4^o,35^o)$
$3^1A'$	6.99	5.09	.231	$+73^o$	6.03	4.9-5.0	.25	(-75^o)
$4^1A'$	7.89	5.96	.023	$+7^o$	5.54	5.7-5.8	$<.05^c$	
$5^1A'$	8.60	6.65	.161	-80^o	10.17	6.1-6.3	.41	$(-71^o,-79^o)$
$6^1A'$	9.76	6.55	.225	-41^o	6.11	6.6-6.7	.48	$(-9^o,41^o)$
$7^1A'$	8.69	6.66	.479	$+43^o$	6.57			
$8^1A'$	9.43	6.77	.098	$+52^o$	7.17			

[a] See ref. [55] for details.

[b] A better match with the experimental values is obtained by considering solvent effects.

due to the presence of the diffuse orbital in the active space. The Rydberg orbital is then removed (placed in the last position of its symmetry and the DELETE option used) from the active space and the calculation repeated. This time the next Rydberg orbital ($3d_{xz}$ or $3d_{yz}$) will take its place. The process is repeated once again until the three Rydberg orbitals have been first included in the active space and then deleted (option DELETED of the RASSCF program). Now we can reduce the active space to (0,10), only including valence orbitals and valence excited states.

We can repeat the calculation including even more roots. The results are in Table 5.16.

The results are quite different from those obtained previously, especially regarding the oscillator strengths and transition dipole moment directions. What we have before was a set of states with valence-Rydberg character, although it was not reflected in the orbital extension or population analysis because the orbitals in the active space were too compact to be able to reflect it. The states we have now are also of clear valence character but the difference is that we have first included the Rydberg orbitals in the active space, allowed the flexibility to describe the Rydberg state, and then removed them from the space to finish with a set of compact valence orbitals which cannot represent the Rydberg states. Then, the latter are removed from the computed spectrum of states.

The experience of this type of treatment in different molecules [43, 52, 55] points out that if the valence states of a molecule are computed without considering the Rydberg states and functions (whether by excluding them from the basis set or from the active space) can result in an additional CASPT2 error as large as 0.3-0.4 eV. The errors are more severe for other transitions properties. One example of this can be found for two different CASPT2 treatments of the formamide molecule, one including diffuse functions and other excluding them (see ref. [56] for details). Notice, however, that this approach cannot describe a true valence-Rydberg mixing. An alternative to such an approach is to use the MultiState

CASPT2 treatment that, although computationally expensive, might properly treats the valence-Rydberg mixing. It must be remembered, however, that the performance of the MS-CASPT2 method relies on the previous mixing of the wave functions, and therefore it will not be unusual, depending on the employed basis set, to obtain CASPT2 results that already give the same answer as MS-CASPT2 results when the initial basis sets are changed.

5.5.3 Other cases.

The calculations become increasingly difficult with increased size of the system or in low symmetry cases. Common problems one has to solve are the selection of the active space when it is not possible to include all orbitals expected to be important and the presence of artificial valence-Rydberg mixing in the description of the states. Specific problems appear in systems containing transition metals, where there are a large amount of states close in energy.

To include all the required orbitals into the active space is sometimes impossible. This is one of the important limitations of the methodology. But some solutions are available if one is aware of the limitations. References [57] and [58] report studies on the porphin and indigo molecules, respectively. Porphin and indigo have 24 and 20 π, π^* orbitals, respectively. It is obviously impossible to include all of them in the active spaces. The analysis of the configurations and occupation numbers of the orbitals in a restricted number of excited states by means of the RASSCF method has been found to be a useful procedure to find a proper active space to study different states of the systems. The RASSCF method is able to deal with a larger number of configurations making possible to include all the π orbitals in the active space and analyze the role of the different orbitals. Our goal in this case is to be able to discard some of the deepest or highest orbitals if they become less important in the description of the desired states.

One possibility is to perform a SDTQ calculation involving all the presumably important active space (occupied orbitals in RAS1, empty orbitals in RAS3, no orbitals in RAS2, and four holes/electrons allowed in RAS1/RAS3). The occupation numbers for the active orbitals obtained for such calculation are usually similar to those of a full CASSCF treatment. Another possibility is to place in the CAS space (RAS2) the most important orbitals and the corresponding electrons and only allow singles and doubles excitations from RAS1 (occupied orbitals) to RAS3 (empty orbitals). In all these cases we will study the configurations and occupation numbers of the orbitals to find if some of them are or minor importance for the description of the states we are considering and then reduce the active space for the CASSCF/CASPT2 calculation [57, 58].

Calculation on the excited states of transition metal compounds have to deal with another set of problems. For instance, the known $3d$ double-shell effect: two sets of d orbitals ($3d$ and $4d$) must be included in the reference space in order to obtain accurate results [43] in molecules containing metal atoms of the first transition row with many d-electrons (Fe-Zn). This is a severe limitation when more ligands are included together with the metal atom. Illustrations of such problems are the calculation of the cyanide and carbonyl transition metal compounds [43, 59] and metal-protein models [60]. Core-valence [61] and relativistic effects [54] have been shown to be important for obtaining accurate results. Finally, the problem of the high multiplicity states in the standard CASPT2 formulation has to be considered. The zeroth-order Hamiltonian is defined as a Fock-type one-electron operator. Apart from the originally proposed Fock matrix [49, 50], a correction, denoted g_1 [62], has been designed so that CASSCF wave functions dominated by a closed-shell configuration, on the one hand, and an open-shell configuration, on the other hand, are treated in similar and balanced ways in the perturbation calculation. This correction was shown to be essential in order to obtain reliable results for the Cr_2 molecule with the CASSCF/CASPT2 method [52].

Each type of system and situation has its own specific problems. Size and convergence problems in systems without any symmetry [63, 64], symmetry breaking and localization problems in high symmetry cases [65], excited states in radical cations [66] and anions [67], etc. In addition, there are situations such as the crossing regions which require the simultaneous treatment of more than one state at the CASPT2 level, which can only be solved using the multi-state option in CASPT2.

5.6 Solvent models.

For isolated molecules of modest size the *ab initio* methods have reached great accuracy at present both for ground and excited states. Theoretical studies on isolated molecules, however, may have limited value to bench chemists since most of the actual chemistry takes place in a solvent. If solute-solvent interactions are strong they may have a large impact on the electronic structure of a system and then on its excitation spectrum, reactivity, and properties. For these reasons, numerous models have been developed to deal with solute-solvent interactions in *ab initio* quantum chemical calculations. A microscopic description of solvation effects can be obtained by a supermolecule approach or by combining statistical mechanical simulation techniques with quantum chemical methods. Such methods, however, demand expensive computations. By contrast, at the phenomenological level, the solvent can be regarded as a dielectric continuum, and there are a number of approaches [68, 69, 70, 71, 72] based on the classical reaction field concept.

MOLCAS can model the solvent within the framework of SCF, RASSCF and CASPT2 programs, for the calculation of energies and properties and also for geometry optimizations. The reaction field formalism is based on a sharp partition of the system: the solute molecule (possibly supplemented by some explicit solvent molecules) is placed in a cavity surrounded by a polarizable dielectric. The surrounding is characterized mainly by its dielectric constant and density: an important parameter of the method is the size of the cavity; the dielectric medium is polarized by the solute, and this polarization creates a reaction field which perturbs the solute itself.

Two versions of the model are presently available: one is based on the Kirkwood model [70, 71] and uses only spherical cavities; the other is called PCM (polarizable continuum model) [68, 69] and can use cavities of general shape, modeled on the actual solute molecule. In the former case, the reaction field is computed as a truncated multipolar expansion and added as a perturbation to the one-electron Hamiltonian; in the latter case the reaction field is expressed in terms of a collection of apparent charges (solvation charges) spread on the cavity surface: the PCM reaction field perturbs both one- and two-electron Hamiltonian operators. In both cases, the solvent effects can be added to the Hamiltonian at any level of theory, including MRCI and CASPT2.

5.6.1 Kirkwood model.

This version of the model only uses spherical cavities. In addition, it includes Pauli repulsion due to the medium by introducing a repulsive potential representing the exchange repulsion between the solute and the solvent. This is done by defining a penalty function of Gaussian type, generating the corresponding spherical well integrals, and adding them to the one-electron Hamiltonian. When the repulsion potential is used, the size of the cavity should be optimized for the ground state of the molecule (see below). If the repulsive potential is not used and the cavity size is chosen to be smaller (molecular size plus van der Waals radius as is the usual choice in the literature) one must be aware of the consequences: larger solvent effects but also an unknown presence of molecular charge outside the boundaries of the cavity. This is not a consequence of the present model but it is a general feature of cavity models [71].

5.6.2 PCM

The cavities are defined as the envelope of spheres centered on solute atoms or atomic groups (usually hydrogens are included in the same sphere of the atoms they are bonded to). Two selection of radii are presently available, i. e. Pauling radii, and the so-called UATM (united atom topological model) radii: the latter is the default for PCM calculations; sphere radii can also be provided by the user in the input file. The solvation charges are placed in the middle of small tiles (*tesserae*) drawn on the surface; the number of solvation charges can be gauged by changing the average area of tesserae (keyword AAre in `SEWARD`).

The program prints some information related to the cavity, where one should always check carefully the magnitude of sphere radii: the program adjusts them automatically to the solute topology (each radius depends on hybridization, bonds, etc.), and sometimes this causes some problems (for instance, discontinuities could appear during the scan of a potential energy surface): if this happens, it is preferable to provide the desired radii in the input file, so that they will be kept at all geometries.

When doing state-average RASSCF calculations, one has to specify which root is to be used to generate the solvation charges: this means that the PCM reaction field will be in equilibrium with a specific electronic state, while it perturbs all the states included in the calculation.

In electronic transitions (e. g. photon absorption or emission) one has to include non-equilibrium effects, due to the finite relaxation time of solvent molecules following a sudden change in electronic distribution. This is done by partitioning the reaction field in two components (fast and slow, the former always equilibrated, the latter delayed), whose magnitude is determined by the static dielectric constant and by a “fast” dielectric constant [73] (for very fast processes, like photon absorption, the fast constant is equal to the square of the refraction index). To perform a non-equilibrium calculation, for example to study a ground-to-excited state transition, one has to perform a regular calculation at equilibrium for the ground state, followed by a calculation for the excited state specifying the keyword NONEQ in the `RASSCF` program. Failing to include the keyword NONEQ will cause the program to compute equilibrium solvation also for the excited state, what would be appropriate for an adiabatic, instead of a vertical, transition.

CASPT2 calculations can be performed as usual for isolated molecules, specifying the keyword RFPERT. Geometry optimizations can be performed as usual: note that the arrangement of solvation charges around the solute molecule is likely to break the molecular symmetry. If the symmetry was explicitly required in `SEWARD`, the system will keep it through the optimization even in the presence of the solvent, otherwise the convergence could be more difficult, and the final geometry could result of a lower symmetry.

5.6.3 Calculation of solvent effects: Kirkwood model.

We begin by performing a CASSCF/CASPT2 reaction field calculation on the ground state of a molecule.

To use the Kirkwood model, the keyword

```
REACtion␣field
```

is needed; if no repulsive potential is going to be used the input simply consists in adding the appropriate data (dielectric constant of the medium, cavity size, and angular quantum number of the highest multipole moment of the charge distribution) into the `SEWARD` input:

```
␣&SEWARD␣&END
...
...
RF-Input
Reaction␣field
80␣8.0␣4
End␣of␣RF-Input
...
...
End␣of␣Input
```

This will compute the reaction field at those levels. The dielectric constant 80.0 correspond to water as solvent. The radius of the cavity is 8.0 in atomic units. Finally 4 is the maximum angular moment number used in the multipole expansion. The cavity origin is the coordinate origin, thus the molecule must be placed accordingly.

If we want to include the reaction field (either PCM or Kirkwood model) at other levels of theory the keyword RFPERT must be added to the `MOTRA` or `CASPT2` inputs.

We are, however, going to explain the more complicated situation where a repulsive well potential has to be added to the model. In this case it is convenient to optimize the size of the cavity, although in so doing we obtain large cavity sizes and therefore smaller solvent effects. More realistic results can be obtained if additional and specific solvent molecules are added inside the cavity.

To define the well potential we have to add the keyword WELL INTEGRALS to the `SEWARD` input to compute and add the Pauli repulsion integrals to the bare Hamiltonian.

The requirements considered to build this potential are that it shall reproduce solvation energies for spherical particles, ions, and that it must be wide enough so that the electrons in the excited state of the molecules are also confined to the cavity. Negative ions have the property that their electrons are loosely bound and they are thus suited for parametrizing the repulsive potential. The final result of different calibration calculations [74, 71] is a penalty function which includes four Gaussians. If a is the radius of the cavity the Gaussians are placed at distances $a + 2.0$, $a + 3.0$, $a + 5.0$ and $a + 7.0$ a.u. from the cavity's center with exponents 5.0, 3.5, 2.0 and 1.4, respectively.

As an example we will use the N,N-dimethylaminobenzonitrile (DMABN) molecule (see Figure 5.13). This is a well known system with large dipole moments both in ground and excited states which suffer important effects due to the polar environment.

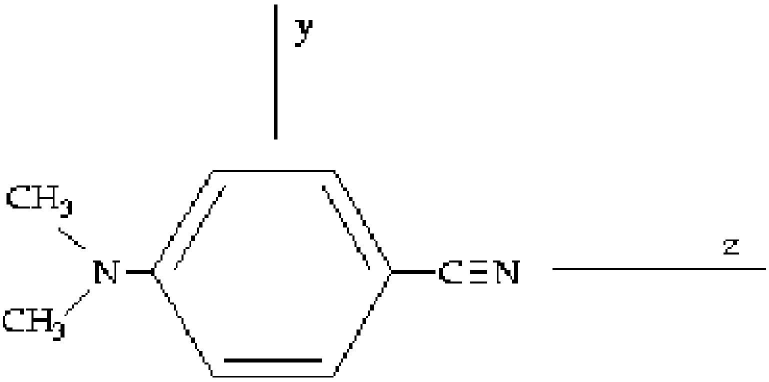


Figure 5.13: N,N-dimethylaminobenzonitrile (DMABN)

```
␣&SEWARD␣&END
Title
para-DMABN␣molecule.␣Cavity␣size:␣10␣au.
Symmetry
␣X␣XY
Basis␣set
N.ANO-S...3s2p1d.
N1␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣4.7847613288
N2␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-8.1106617786
End␣of␣basis
Basis␣set
C.ANO-S...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.1618352923
C2␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.2430930886␣␣␣␣␣␣␣␣0.7747833630
C3␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.2317547910␣␣␣␣␣␣␣-1.8500321252
C4␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-3.1917306021
C5␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-5.9242601761
C6␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.4377336900␣␣␣␣␣␣␣␣6.0640991723
End␣of␣basis
Basis␣set
H.ANO-S...2s.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣4.0043085530␣␣␣␣␣␣␣-2.8534714086
H2␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣4.0326542950␣␣␣␣␣␣␣␣1.7215314260
H3␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.1467175630␣␣␣␣␣␣␣␣8.0879851846
H4␣␣␣␣␣␣␣␣␣␣␣␣␣1.5779129980␣␣␣␣␣␣␣␣3.6622699270␣␣␣␣␣␣␣␣5.5104123361
End␣of␣basis
RF-Input
reaction␣field
38.8␣10.0␣4
End␣of␣RF-Input
Well␣Int
4
1.0␣5.0␣12.0
1.0␣3.5␣13.0
1.0␣2.0␣15.0
1.0␣1.4␣17.0
End␣of␣Input
␣&SCF␣&END
TITLE
␣DMABN␣molecule
OCCUPIED
20␣2␣12␣5
ITERATIONS
```

```
50
END␣OF␣INPUT
␣&RASSCF␣&END
TITLE
␣p-DMABN
SYMMETRY
␣␣␣␣1
SPIN
␣␣␣␣1
NACTEL
␣␣␣10␣␣␣␣0␣␣␣␣0
FROZEN
␣␣␣␣8␣␣␣␣0␣␣␣␣3␣␣␣␣0
INACTIVE
␣␣␣12␣␣␣␣1␣␣␣␣9␣␣␣␣1
RAS2
␣␣␣␣0␣␣␣␣2␣␣␣␣0␣␣␣␣7
THRS
1.0E-06,1.0E-03,1.0E-03
ITER
50,25
LUMORB
END␣OF␣INPUT
```

In the `SEWARD` input the WELL INTEGRALS must include first the number of Gaussians used (four), followed by the coefficient and exponent of the Gaussian and the radius of the cavity in the sequence explained above: first the most compact Gaussian with the radius plus 2.0 au, and so on to the least compact Gaussian. Here, we have defined a cavity size of 10 au (cavity centered at coordinate origin). The RASSCF program will read the RCTFLD input, prepared this time for acetonitrile ($\epsilon = 38.8$), a cavity size of 10.0 au (the same as in the SEWARD input) and a multipole expansion up to the fourth order which is considered sufficient [71]. The active space includes the π space over the molecular plane, excluding the π orbital of the CN group which lies in the molecular plane.

We repeat the calculation for different cavity sizes in order to find the radius which gives the lowest absolute energy at the CASSCF level. The presence of the repulsive terms allows the cavity radius to be computed by energy minimization. For the calculations using different cavity sizes it is not necessary to repeat the calculation of all the integrals, just those related to the well potential. Therefore, the keyword ONEONLY can be included in the SEWARD input. The *`ONEINT`* file will be modified and the *`ORDINT`* file is kept the same for each molecular geometry. The energies obtained are in Table 5.17.

Radius (au)	CASSCF energies (au)
no cav.	-455.653242
10.0	-455.645550
11.0	-455.653486
12.0	-455.654483
14.0	-455.654369
16.0	-455.654063

Table 5.17: Ground state CASSCF energies for DMABN with different cavity sizes.

Taking the gas-phase value (no cav.) as the reference, the CASSCF energy obtained with a 10.0 au cavity radius is higher. This is an effect of the repulsive potential, meaning that the molecule is too close to the boundaries. Therefore we discard this value and use the values from 11.0 to 16.0 to make a simple second order fit and obtain a minimum for the cavity radius at 13.8 au.

Once we have this value we also need to optimize the position of the molecule in the cavity. Some parts of the molecule, especially those with more negative charge, tend to move close to the boundary. Remember than the sphere representing the cavity has its origin in the cartesian coordinates origin. We use the radius of 13.8 au and compute the CASSCF energy at different displacements along the coordinate axis. Fortunately enough, this molecule has C_{2v} symmetry. That means that displacements along two of the axis (x and y) are restricted by symmetry. Therefore it is necessary to analyze only the displacements along the z coordinate. In a less symmetric molecule all the displacements should be studied even including combination of the displacements. The result may even be a three dimensional net, although no great accuracy is really required. The results for DMABN n C_{2v} symmetry are compiled in Table 5.18.

Disp. in z (au)	CASSCF energies (au)
+0.5	-455.654325
0.0	-455.654400
-0.5	-455.654456
-1.0	-455.654486
-1.5	-455.654465

Table 5.18: Ground state CASSCF energies for different translations with respect to the initial position of of the DMABN molecule in a 13.8 au cavity.

Fitting these values to a curve we obtain an optimal displacement of -1.0 au. We move the molecule and reoptimize the cavity radius at the new position of the molecule. The results are listed in Table 5.19.

Radius (au)	CASSCF energies (au)
11.8	-455.653367
12.8	-455.654478
13.8	-455.654486
14.8	-455.654318

Table 5.19: Ground state CASSCF energies for DMABN with different cavity sizes. The molecule position in the cavity has been optimized.

There is no significant change. The cavity radius is then selected as 13.8 au and the position of the molecule with respect to the cavity is kept as in the last calculation. The calculation is carried out with the new values. The SCF or RASSCF outputs will contain the information about the contributions to the solvation energy. The CASSCF energy obtained will include the reaction field effects and an analysis of the contribution to the solvation energy for each value of the multipole expansion:

```
Reaction field specifications:
------------------------------
```

```
 Dielectric Constant :                .388E+02
 Radius of Cavity(au):                .138E+02
 Truncation after    :               4

Multipole analysis of the contributions to the dielectric solvation energy

 -------------------------------------
    l            dE
 -------------------------------------
    0           .0000000
    1          -.0013597
    2          -.0001255
    3          -.0000265
    4          -.0000013
 -------------------------------------
```

5.6.4 Solvation effects in ground states. PCM model in formaldehyde.

The reaction field parameters are added to the `SEWARD` program input through the keyword

```
RF-Input
```

To invoke the PCM model the keyword

```
PCM-model
```

is required. A possible input is

```
RF-input
PCM-model
solvent
acetone
AAre
0.2
End␣of␣rf-input
```

which requires a PCM calculation with acetone as solvent, with tesserae of average area 0.2 Å^2. Note that the default parameters are solvent=water, average area 0.4 Å^2; see the `SEWARD` manual section for further PCM keywords. By default the PCM adds non-electrostatic terms (i. e. cavity formation energy, and dispersion and repulsion solute-solvent interactions) to the computed free-energy in solution.

A complete input for a ground state CASPT2 calculation on formaldehyde (H_2CO) in water is

```
␣&SEWARD␣&END
Title
formaldehyde
Symmetry
␣x␣y
Basis␣set
H.6-31G*....
H1␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.924258␣␣␣-1.100293␣/Angstrom
End␣of␣basis
Basis␣set
C.6-31G*....
C3␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣-0.519589␣/Angstrom
End␣of␣basis
Basis␣set
```

```
O.6-31G*....
O␣␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣0.664765␣/Angstrom
End␣of␣basis
RF-input
PCM-model
solvent
water
end␣of␣rf-input
End␣of␣input
␣&SCF␣&END
Title
formaldehyde
ITERATIONS
50
Occupied
5␣1␣2␣0
End␣of␣input
␣&RASSCF␣&END
Title
formaldehyde
nActEl
4␣0␣0
Symmetry
1
Inactive
4␣0␣2␣0
Ras2
1␣2␣0␣0
CiRoot
1␣1
1
Iter
100,20
LumOrb
End␣of␣input
␣&CASPT2␣&END
Title
formaldehyde
MaxIterations
20
Frozen
4
RFPErt
End␣of␣input
```

If a $n \rightarrow \pi^*$ electronic transition is to be studied at the RASSCF level, the ground state is computed with the same `SEWARD`, `SCF` and `RASSCF` inputs as above, while the excited state is computed with the following `RASSCF` input:

```
␣&RASSCF␣&END
Title
formaldehyde
nActEl
4␣0␣0
Symmetry
4
Inactive
5␣0␣1␣0
Ras2
0␣2␣1␣0
CiRoot
1␣1
1
Iter
100,20
```

```
JOBIPH
NONEQ
End␣of␣input
```

Note the PCM keyword NONEQ, requiring that the slow part of the reaction field be frozen as in the ground state, while the fast part is equilibrated to the new electronic distribution. In this case the fast dielectric constant is the square of the refraction index, whose value is tabulated for all the allowed solvents (anyway, it can be modified by the user through the keyword "INFInite" in `SEWARD`).

5.6.5 Solvation effects in excited states. PCM model and acrolein.

In the PCM picture, the solvent reaction field is expressed in terms of a polarization charge density $\sigma(\mathbf{s})$ spread on the cavity surface, which, in the most recent version of the method, depends on the electrostatic potential $V(\mathbf{s})$ generated by the solute on the cavity according to

$$\left[\frac{\epsilon+1}{\epsilon-1}\hat{S} - \frac{1}{2\pi}\hat{S}\hat{D}^*\right]\sigma(\mathbf{s}) \quad = \quad \left[-1+\frac{1}{2\pi}\hat{D}\right]V(\mathbf{s}) \tag{5.5}$$

where ϵ is the solvent dielectric constant and $V(\mathbf{s})$ is the (electronic+nuclear) solute potential at point $\mathbf{s}$ on the cavity surface. The $\hat{S}$ and $\hat{D}^*$ operators are related respectively to the electrostatic potential $V^{\sigma}(\mathbf{s})$ and to the normal component of the electric field $E^{\sigma}_{\perp}(\mathbf{s})$ generated by the surface charge density $\sigma(\mathbf{s})$. It is noteworthy that in this PCM formulation the polarization charge density $\sigma(\mathbf{s})$ is designed to take into account implicitly the effects of the fraction of solute electronic density lying outside the cavity.

In the computational practice, the surface charge distribution $\sigma(\mathbf{s})$ is expressed in terms of a set of point charges $\mathbf{q}$ placed at the center of each surface tessera, so that operators are replaced by the corresponding square matrices. Once the solvation charges ($\mathbf{q}$) have been determined, they can be used to compute energies and properties in solution.

The interaction energy between the solute and the solvation charges can be written

$$E_{int} = \mathbf{V}^{\dagger}\mathbf{q} = \sum_{\mathbf{i}}^{\mathbf{N_{TS}}} \mathbf{V_i q_i} \tag{5.6}$$

where V_i is the solute potential calculated at the representative point of tessera i. The charges act as perturbations on the solute electron density ρ: since the charges depend in turn on ρ through the electrostatic potential, the solute density and the charges must be adjusted until self consistency. It can be shown[72] that for any SCF procedure including a perturbation linearly depending on the electron density, the quantity that is variationally minimized corresponds to a free energy (i.e. E_{int} minus the work spent to polarize the dielectric and to create the charges). If $E^0 = E[\rho^0] + V_{NN}$ is the solute energy in vacuo, the free energy minimized in solution is

$$\mathcal{G} = E[\rho] + V_{NN} + \frac{1}{2}E_{int} \tag{5.7}$$

where V_{NN} is the solute nuclear repulsion energy, ρ^0 is the solute electronic density for the isolated molecule, and ρ is the density perturbed by the solvent.

The inclusion of non-equilibrium solvation effects, like those occurring during electronic excitations, is introduced in the model by splitting the solvation charge on each surface element into two components: $q_{i,f}$ is the charge due to electronic (fast) component of solvent polarization, in equilibrium with the solute electronic density upon excitations, and $q_{i,s}$, the charge arising from the orientational (slow) part, which is delayed when the solute undergoes a sudden transformation.

The photophysics and photochemistry of acrolein are mainly controlled by the relative position of the $^1(n-\pi^*)$, $^3(n-\pi^*)$ and $^3(\pi-\pi^*)$ states, which is, in turn, very sensitive to the presence and the nature of the solvent. We choose this molecule in order to show an example of how to use the PCM model in a CASPT2 calculation of vertical excitation energies.

The three states we want to compute are low-lying singlet and triplet excited states of the *s-trans* isomer. The π space (4 π MOs /4 π-electrons) with the inclusion of the lone-pair MO (n_y) is a suitable choice for the active space in this calculation. For the calculation in aqueous solution, we need first to compute the CASPT2 energy of the ground state in presence of the solvent water. This is done by including in the `SEWARD` input for the corresponding gas-phase calculation the section

```
RF-input
PCM-model
solvent
␣water
DIELectric␣constant
␣78.39
CONDuctor␣version
AARE
␣0.4
End␣of␣rf-input
```

If not specified, the default solvent is chosen to be water. Some options are available. The value of the dielectric constant can be changed for calculations at temperatures other than 298 K. For calculations in polar solvents like water, the use of the conductor model (C-PCM) is recommended. This is an approximation that employs conductor rather than dielectric boundary conditions. It works very well for polar solvents (i. e. dielectric constant greater than about 5), and is based on a simpler and more robust implementation. It can be useful also in cases when the dielectric model shows some convergence problems. Another parameter that can be varied in presence of convergency problem is the average area of the tesserae of which the surface of the cavity is composed. However, a lower value for this parameter may give poorer results.

Specific keywords are in general needed for the other modules to work with PCM, except for the SCF. The keyword NONEQUILIBRIUM is necessary when computing excited states energies in `RASSCF`. For a state specific calculation of the ground state CASSCF energy, the solvent effects must be computed with an equilibrium solvation approach, so this keyword must be omitted. None the less, the keyword RFPERT must be included in the CASPT2 input in order to add the reaction field effects to the one-electron hamiltonian as a constant perturbation.

```
␣&RASSCF␣&END
Title
Acrolein␣GS␣+␣PCM
Spin
␣1
Symmetry
```

```
␣1
nActEl
␣6␣0␣0
Frozen
␣4␣0
Inactive
␣8␣0
Ras2
␣1␣4
LUMORB
THRS
1.0e-06␣1.0e-04␣1.0e-04
ITERation
␣100␣100
End␣of␣input
␣&CASPT2␣&END
Title
␣ground␣state␣+␣PCM
RFpert
End␣of␣Input
```

Information about the reaction field calculation employing a PCM-model appear first in the SCF output

```
Polarizable Continuum Model (PCM) activated
Solvent:water
Version: Conductor
Average area for surface element on the cavity boundary: 0.4000 Angstrom2
Minimum radius for added spheres: 0.2000 Angstrom
Polarized Continuum Model Cavity
================================
 Nord Group  Hybr  Charge Alpha Radius     Bonded to
   1   O     sp2   0.00   1.20  1.590   C   [d]
   2   CH    sp2   0.00   1.20  1.815   O   [d]  C   [s]
   3   CH    sp2   0.00   1.20  1.815   C   [s]  C   [d]
   4   CH2   sp2   0.00   1.20  2.040   C   [d]
 ------------------------------------------------------------------------------
```

The following input is used for the CASPT2 calculation of the ${}^3A''(n \rightarrow \pi^*)$ state. Provided that the same $WorkDir has been using, which contains all the files of of the calculation done for the ground state, the excited state calculation is done by using inputs for the `RASSCF` and the `CASPT2` calculations:

```
 &RASSCF &END
Title
Acrolein n->pi* triplet state + PCM
Spin
 3
Symmetry
 2
nActEl
 6 0 0
Frozen
 4 0
Inactive
 8 0
Ras2
 1 4
NONEquilibrium
LUMORB
ITERation
 100 100
End of input
```

```
 &CASPT2 &END
Title
 triplet state
RFpert
End of Input
```

The `RASSCF` output include the line:

```
Reaction field from state:            1
```

This piece of information means that the program computes the solvent effects on the energy of the $^3A''(n \rightarrow \pi^*)$ by using a non-equilibrium approach. The slow component of the solvent response is kept frozen in terms of the charges that have been computed for the equilibrium calculation of the ground state (state 1). The remaining part of the solvent response, due to the fast charges, is instead computed self-consistently for the state of interest.

The vertical excitations to the lowest valence states in aqueous solution for *s-trans* acrolein are listed in the Table 5.20 and compared with experimental data. As expected by qualitative reasoning, the vertical excitation energy to the $^1A''(n \rightarrow \pi^*)$ state exhibits a blue shift in water. The value of the vertical transition energy computed with the inclusion of the PCM reaction field is computed to be 3.96 eV at the CASPT2 level of theory. The solvatochromic shift is thus of +0.33 eV. Experimental data are available for the excitation energy to the $^1A''(n \rightarrow \pi^*)$ state. The band shift in going from isooctane to water is reported to be +0.24 eV which is in fair agreement with the PCM result.

No experimental data are available for the excitation energies to the triplet states of acrolein in aqueous solution. However it is of interest to see how the ordering of these two states depends on solvent effects. The opposing solvatochromic shifts produced by the solvent on these two electronic transitions place the two triplet states closer in energy. This result might suggest that a dynamical interconversion between the $n\pi^*$ and $\pi\pi^*$ may occur more favorable in solution.

Table 5.20: Vertical excitation energies/eV (solvatochromic shifts) of s-trans acrolein in gas-phase and in aqueous solution

State	Gas-phase	Water	Expt.[a]
$^1A''(n_y \rightarrow \pi^*)$	3.63	3.96 (+0.33)	3.94 (+0.24)[b]
T_1 $^3A''(n_y \rightarrow \pi^*)$	3.39	3.45 (+0.06)	
T_2 $^3A'(\pi \rightarrow \pi^*)$	3.81	3.71 (-0.10)	

[a] Ref.[75]

[b] Solvatochromic shifts derived by comparison of the absorption wave lengths in water and isooctane

5.7 Computing relativistic effects in molecules.

MOLCAS-7.4 is intended for calculations on systems including all atoms of the periodic table. This is only possible if relativistic effects can be added in a way that is accurate and at the same time applies to all the methods used in *MOLCAS*-7.4, in particular the CASSCF and CASPT2 approaches. *MOLCAS*-7.4 includes relativistic effects within the same wave function framework as used in non-relativistic calculations. This has been possible by partitioning the relativistic effects into two parts: the scalar relativistic effects and spin-orbit coupling. This partitioning is based on the Douglas-Kroll (DK) transformation of the relativistic Hamiltonian [76, 77].

5.7.1 Scalar relativistic effects

The scalar relativistic effects are included by adding the corresponding terms of the DK Hamiltonian to the one-electron integrals in Seward (use the keyword DOUGLAS-KROLL). This has no effect on the form of the wave function and can be used with all *MOLCAS*-7.4 modules. Note however that it is necessary to use a basis set with a corresponding relativistic contraction. *MOLCAS*-7.4 provides the ANO-RCC basis set, which has been constructed using the DK Hamiltonian. Use this basis set in your relativistic calculations. It has the same accuracy as the non-relativistic ANO-L basis set. Scalar relativistic effects become important already for atoms of the second row. With ANO type basis sets it is actually preferred to use the DK Hamiltonian and ANO-RCC in all your calculations.

5.7.2 Spin-Orbit coupling (SOC)

In order to keep the structure of *MOLCAS* as intact as possible, it was decided to incorporate SOC as an *a posteriori* procedure which can be added after a series of CASSCF calculations. The program `RASSI` has been modified to include the spin-orbit part of the DK Hamiltonian [78]. The method is thus based on the concept of electronic states interacting via SOC. In practice this means that one first performs a series of CASSCF calculations in the electronic states one expects to interact via SOC. They are then used as the basis states in the RASSI calculations. Dynamic electron correlation effects can be added by a shift of the diagonal of the SOC Hamiltonian to energies obtained in a CASPT2 or MRCI calculation. If MS-CASPT2 is used, a special output file (*`JOBMIX`*) is provided that is to be used as the input file for RASSI. The procedure will below be illustrated in a calculation on the lower excited states of the PbO molecule.

The SO Hamiltonian has been approximated by a one-electron effective Hamiltonian [79], which also avoids the calculation of multi-center integrals (the Atomic Mean Field Approximation – AMFI) [79, 80].

5.7.3 The PbO molecule

Results from a calculation of the potentials for the ground and lower excited states of PbO, following the procedure outlined above, has recently been published [81]. The ground state of PbO dissociates to O(^3P) and Pb(^3P). However in the Pb atom there is strong SOC between the ^3P, ^1D, and ^1S term of the $(6s)^2(6p)^2$ electronic configuration. All levels with the Ω value O^+ arising from these terms will therefore contribute to the ground state potential. The first task is therefore to construct the electronic states that are obtained by coupling O(^3P) to any of the ^3P, ^1D, and ^1S terms of Pb. In the table below we give the states. They have been labeled both in linear symmetry and in C_2 symmetry, which is the symmetry used in the calculation because it makes it possible to average over degenerate components.

Spin	C_2 sym	Labels in linear symmetry	No. of states
2	1	$^5\Delta$, $2 \times {}^5\Sigma^+$, $^5\Sigma^-$	5
2	2	$2 \times {}^5\Pi$	4
1	1	$3 \times {}^3\Delta$, $3 \times {}^3\Sigma^+$, $4 \times {}^3\Sigma^-$	13
1	2	$6 \times {}^3\Pi$, $^3\Phi$	14
0	1	$^1\Delta$, $2 \times {}^1\Sigma^+$, $^1\Sigma^-$	5
0	2	$2 \times {}^1\Pi$	4

The total number of states is 45. One thus has to perform 6 CASSCF (and MS-CASPT2) calculations according to the spin and symmetries given in the table. The RASSI-SO calculation will yield 134 levels with Ω ranging from 0 to 4. Only the lower of these levels will be accurate because of the limitations in the selection of electronic states.

The active space used in these calculations is 6s,6p for Pb and 2p for O. This is the natural choice and works well for all main group elements in most molecules. The s-orbital should be active in groups IIa-Va, but may be left inactive for the heavier atoms (groups VIa-VIIa). The ANO-RCC basis sets have been constructed to include correlation of the semi-core electrons. For Pb they are the 5d, which should then not be frozen in the `CASPT2` calculations. All other core electrons should be frozen, because there are no basis functions to describe their correlation. Including them in the correlation treatment may lead to large BSSE errors.

The input file for these calculations is quite lengthy, so we show here only one set of `CASSCF/CASPT2` calculations but the whole `RASSI` input for all six cases.

```
************************************************************************
␣&SEWARD␣␣&END
Title
␣PbO
Symmetry
␣XY
Douglas-Kroll
Amfi
Basis␣set
Pb.ano-rcc.Roos.25s22p16d12f4g.9s8p6d4f3g.
Pb␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣0.000000
End␣of␣basis
Basis␣set
O.ano-rcc.Roos.14s9p4d3f2g.5s4p3d2f1g.
O␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣Dist
End␣of␣basis
End␣of␣input
************************************************************************
!ln␣-fs␣$CurrDir/ScfOrb␣SCFORB
␣&SCF␣&END
Title
␣PbO
Occupied
␣24␣21
Iterations
␣20
Prorbitals
␣␣␣2␣1.d+10
End␣of␣input
************************************************************************
!ln␣-fs␣$CurrDir/ScfOrb␣INPORB
!ln␣-fs␣$CurrDir/JobIph.12␣JOBIPH
!ln␣-fs␣$CurrDir/JobOld.12␣JOBOLD
␣&RASSCF␣&END
Title
␣PbO
```

```
Symmetry
␣1
Spin
␣5
nActEl
␣8␣0␣0
Inactive
␣23␣18
Ras2
␣3␣4
*Lumorb
JobIph
THRS
1.0e-8␣1.0e-04␣1.0e-04
Levshft
␣1.50
ITERation
200␣50
CIMX
␣200
CIROOT
␣5␣5␣1
SDAV
␣500
End␣of␣input
!cp␣$CurrDir/JobIph.12␣$CurrDir/JobOld.12
************************************************************************
!ln␣-fs␣$CurrDir/JobMix.12␣JOBMIX
␣&CASPT2␣&END
Title
␣PbO
MAXITER
␣25
FROZEN
␣19␣16
Focktype=G1
Multistate
␣5␣1␣2␣3␣4␣5
Imaginary␣Shift
␣0.1
End␣of␣input
************************************************************************
!ln␣-fs␣$CurrDir/JobMix.12␣JOB001
!ln␣-fs␣$CurrDir/JobMix.11␣JOB002
!ln␣-fs␣$CurrDir/JobMix.21␣JOB003
!ln␣-fs␣$CurrDir/JobMix.10␣JOB004
!ln␣-fs␣$CurrDir/JobMix.22␣JOB005
!ln␣-fs␣$CurrDir/JobMix.20␣JOB006
␣&RASSI␣&END
Nrof␣JobIphs
␣6␣5␣13␣14␣5␣4␣4
␣1␣2␣3␣4␣5
␣1␣2␣3␣4␣5␣6␣7␣8␣9␣10␣11␣12␣13
␣1␣2␣3␣4␣5␣6␣7␣8␣9␣10␣11␣12␣13␣14
␣1␣2␣3␣4␣5
␣1␣2␣3␣4
␣1␣2␣3␣4
Spin␣Orbit
Ejob
End␣of␣input
************************************************************************
```

In the above definitions of the JobMix files the labels correspond to symmetry and spin. Thus *JobMix.12* is for quintets (S=2) in symmetry 1, etc. The keyword EJOB ensures that the `MS-CASPT2` energies from the *JobMix* files are used as the diagonal elements in the SO Hamiltonian matrix. The output file of one such calculation is quite lengthy (6 `CASSCF/MS-CASPT2` calculations and one `RASSI`). Important sections of the `RASSI` output are the spin-free energies (look for the word "SPIN-FREE" in the listing) and the SOC energies (found by looking for "COMPLEX"). The complex SO wave functions are also given and can be used to analyze the wave function. For linear molecules one wants to know the Ω values of the different solutions. Here the computed transition moments can be quite helpful (using the selection rules). It is important in a calculation of many excited states, as the one above, to check for intruder state problems in the `CASPT2` results.

This example includes a large number of states, because the aim was to compute full potential curves. If one is only interested in the properties near equilibrium, one can safely reduce the number of states. For lighter atoms it is often enough to include the spin-free states that are close in energy in the calculation of the SOC. An intersystem crossing can usually be treated by including only the two crossing states. The choice of basis states for the `RASSI` calculation depends on the strength of the SO interaction and the energy separation between the states.

The above input is for one distance. The shell script loops over distances according to:

```
Dist='50.0␣10.0␣8.00␣7.00␣6.00␣5.50␣5.00␣4.40␣4.20␣4.00␣3.90␣3.80␣3.75␣3.70
3.65␣3.60␣3.55␣3.50␣3.40␣3.30␣3.10'
␣for␣R␣in␣$Dist
␣do
␣cat␣$CurrDir/template␣|␣sed␣-e␣"s/Dist/$R/"␣>$CurrDir/input
rm␣-rf␣$WorkDir
mkdir␣␣$WorkDir
cd␣␣␣␣␣$WorkDir
echo␣"R=$R"␣>>$CurrDir/energies
molcas␣$CurrDir/input␣>$CurrDir/out_$R
grep␣"Reference␣energy"␣$CurrDir/out_$R␣>>$CurrDir/energies
grep␣"Total␣energy"␣$CurrDir/out_$R␣>>$CurrDir/energies
grep␣"Reference␣weight"␣$CurrDir/out_$R␣>>$CurrDir/energies
done
```

Thus, the whole potential curves can be run as one job (provided that there are no problems with intruder states, convergence, etc). Notice that the *`JOBIPH`* files for one distance are used as input (*`JOBOLD`*) for the next distance. The shell script collects all `CASSCF` and `CASPT2` energies and reference weights in the file *`energies`*.

We shall not give any detailed account of the results obtained in the calculation of the properties of the PbO molecule. The reader is referred to the original article for details [81]. However it might be of interest to know that the computed dissociation energy (D_0) was 5.0 eV without SOC and 4.0 eV with (experiment is 3.83 eV). The properties at equilibrium are much less affected by SOC: the bond distance is increased with 0.003 Å, the frequency is decreased with 11 cm^{-1}. The results have also been used to assign the 10 lowest excited levels.

Section 6

Acknowledgment

The authors of the *MOLCAS* 7.4 tutorials and examples manual would like to acknowledge all *MOLCAS* co-authors their contributions, suggestions, and proof reading of the present manual. There are many persons whose experience with *MOLCAS* has been a valuable source of information for us. We thank all of them.

We expect this manual to be a useful tool to all *MOLCAS* users. There will be many things to correct, add, and improve. All comments and suggestion will be deeply appreciated.

Part II

User's Guide

Section 7

The *MOLCAS* environment

This section describes how to use the *MOLCAS* program system. The reader is assumed to be familiar with the workings of the operating system, and only issues that are *MOLCAS* specific will be covered.

7.1 Overview

MOLCAS contains a set of ab initio quantum chemical programs. These programs are essentially separate entities, but they are tied together by a shell. The exchange of information between the programs is through files. The shell is designed to allow ease of use with a minimum amount of specifications in a 'run of the mill' case. The shell is flexible and allows the user to perform any calculation possible within the limitations of the various codes supplied with *MOLCAS*.

To make a calculation using *MOLCAS* you have to decide on which programs you need to use, prepare input for these, and construct a command procedure file to run the various programs. This command procedure file is submitted for batch execution. The following two subsections describe the programs available and the files used in *MOLCAS*.

7.1.1 Programs in the system

Below is a list of the available programs given. The programs are tied together with a shell and the inter-program information is passed through files. These files are also specified in this list to indicate the program module interdependencies.

Program	*Purpose*
`ALASKA`	This program computes the first derivatives of the one- and two-electron integrals with respect to the nuclear positions. The derivatives are not stored on files, but contracted immediately with the one- and two-electron densities to form the molecular gradients.

`CASPT2`	This program computes the second order Many Body Perturbation Theory correction to a CASSCF wave function.
`CASVB`	This program performs various types of valence bond calculations. It may be called directly (for VB interpretation of CASSCF wave functions), or within the RASSCF program (for fully variational VB calculations). In the former case it requires the information in the *`JOBIPH`* file generated by the RASSCF program, possibly also the integral files ONEINT and ORDINT.
`CCSDT`	This is a shell-script but is implemented into the *MOLCAS* framework as if it is a program. It contains all logics required to perform a coupled cluster calculation and will invoke the programs `CCSORT`, `CCSD`, and `CCT3`
`CPF`	This program produces a CPF, MCPF or ACPF wave function from a single reference configuration.
`EXPBAS`	This program takes one orbital file generated with a smaller basis set (ANO) and expands it to a larger basis set.
`GATEWAY`	This program collects all information about computed system, including geometry, basis sets and symmetry, and stores the data for a future use.
`GENANO`	This program is used to construct ANO type basis sets.
`GRID_IT`	This program calculates densities and molecular orbitals in a set of cartesian grid points, and produce a file for visualisation of MO's and densities.
`GV`	GUI code for visualization and manipulation of molecular structures, and for visualization of molecular orbitals, densities, density differences, etc.
`FFPT`	This program applies perturbations to the one-electron Hamiltonian for finite field perturbation calculations.
`GUGA`	This program generates the coupling coefficients required by the `MRCI` and `CPF` programs.
`MCKINLEY`	This program calculates the second and first order derivatives of integrals that are used for calculating second order derivatives of the energies with perturbation dependent basis sets.
`MBPT2`	This program computes the second order Many Body Perturbation Theory correction to an SCF wave function.
`MCLR`	This program calculates the response of the wave function and related second order properties.
`MOTRA`	This program transforms one- and two-electron integrals from AO basis to MO basis. The integrals that are transformed are the one-electron Hamiltonian and the two-electron repulsion integrals.

`MRCI`	This program produces a Multi Reference CI wave function from an arbitrary set of reference configurations. Alternatively the program can produce an Averaged CPF wave function.
`MULA`	This program calculates intensities of vibrational transitions between electronic states.
`RASSCF`	This program generates RASSCF type wave functions.
`RASSI`	This program computes the interaction between several RASSCF wave functions. A spin-orbit Hamiltonian can be used.
`SCF`	This program generates Closed Shell SCF or Unrestricted SCF wave functions. Optionally, the calculations can be carried out in 'direct' fashion.
`SEWARD`	This program generates one- and two-electron integrals needed by other programs. If requested the two-electron integrals are computed in the form of Cholesky decomposed vectors.
`SLAPAF`	This program is a general purpose facility for geometry optimization, transition state search, MEP, conical intersections, intersystem crossingd, etc. using analytical or numerical gradients produced by `ALASKA`.
`VIBROT`	This program computes the vibrational-rotational spectrum of a diatomic molecule. Spectroscopic constants are computed. The program can also compute transition moments, life times , etc for excited state potentials.

7.2 Commands and environment variables

This section will describe the usage of MOLCAS in an UNIX environment.

Production jobs using MOLCAS in an UNIX environment can be performed as batch jobs. This requires the creation of a shell script that contains a few simple commands. Further you need to create input for each program module that you intend to use. This section describes the necessary steps you have to take in order to make a successful job using MOLCAS. Input examples for a typical MOLCAS run can be found in `doc/samples/problem_based_tutorials/` directory. Also you can use some input examples in `Test/input` subdirectory.

7.2.1 Commands

There is a command supplied with the MOLCAS package, named molcas, that the user issue to perform a given task. A sequence of such commands will perform the calculation requested by the user.

Command	*Purpose*

molcas	This command tells which molcas installation will be used, and gives some help about usage of molcas command
molcas *`input-file`*	This command executes a command in the MOLCAS system.
molcas help `PRGM`	This command gives the list of available keywords for program `PRGM`.
molcas help `PRGM KEYWORD`	This command gives description of a `KEYWORD`.
molcas help `ENVIRONMENT`	This command gives a list of MOLCAS specific environment variables.
molcas help `BASIS ELEMENT`	This command gives a list of basis sets available for an `ELEMENT`.

The following is an example of running molcas by using a single input file:

```
molcas $Project.input
```

An alternative way of running molcas as a sequence of separate calls:

```
molcas $Project.seward.input    # Execute seward
molcas $Project.scf.input       # Execute scf
```

The default behavor of molcas execution can be altered by setting environment variables.

7.2.2 Project name and working directory

When running a project, MOLCAS uses the variable PROJECT giving a project name, and a scratch directory defined by the variable WORKDIR. This serves the purpose of maintaining structure of the files and facilitating automatic file mapping.

There are several ways to set up these variables. By default, the name of the Project constructed from the name of the input file, by removing the last suffix, e.g. for example for an input name ***`Water.SCF.input`*** the PROJECT name will be WATER.SCF. Alternatively, user can set environment variable PROJECT, or MOLCAS_PROJECT.

Scratch directory can be set by environment variable MOLCAS_WORKDIR. If it is set to value ”PWD”, current directory will be used. Otherwise, it can be set to a directory name. In this case scratch area will be located in a subdirectory $MOLCAS_WORKDIR/$Project. It is also possible to overwrite the value of scratch area, by setting environment variable WORKDIR.

- Project=...; export Project
- WorkDir=...; export WorkDir

MOLCAS modules communicates between each other via files, located in the WORKDIR. The description of internal filenames and file mapping can be found at Appendix.

7.2.3 Input

When you have decided which program modules you need to use to perform your calculation, you need to construct input for each of these. There is no particular structure enforced on the input files, but it is recommended that you follow:

- *$Project.*“*prgm-name*”*.input*

which is the name of the input files assumed in the sample shell script.

7.2.4 Preparing a job

When you prepare a job for batch processing, you have to create a shell script. It is recommended that you use the sample shell script supplied with *MOLCAS* as a starting point when building your own shell script. The following steps are taken in the shell script:

1. Define and export the *MOLCAS* variables
    - Project (or use $MOLCAS\_PROJECT$)
    - WorkDir (or $MOLCAS\_WORKDIR$)
2. Issue a sequence of *MOLCAS* commands.
3. Remove the scratch directory and all files in it.

The following is an example of a shell script.

```
Project=HF; export Project                             # Define the project id
WorkDir=/temp/$LOGNAME/$Project.$RANDOM; export WorkDir  # Define scratch directory
molcas $Project.input                                  # Run molcas with input file, which
                                                       #   contains inputs for several modules
rm -r $WorkDir                                         # Clean up
```

The file *$ThisDir/$Project.input* contains the ordered sequence of *MOLCAS* inputs and `AUTO` will call the appropriate programs. See section 8.9 for an explanation of the additional tools available in the `AUTO` program.

The following is an example of a shell script to be submitted for batch execution.

```
Project=HF; export Project                             # Define the project id
WorkDir=/temp/$LOGNAME/$Project.$RANDOM; export WorkDir  # Define scratch directory
molcas $Project.seward.input                           # Execute seward
molcas $Project.scf.input                              # Execute scf
rm -r $WorkDir                                         # Clean up
```

Note, that if you configured your working environment by using `setuprc` script, the only command you have to place into the shell script is:

```
molcas $Project.input
```

7.2.5 System variables

MOLCAS contains a set of system variables that the user can set to modify the default behaviour of *MOLCAS*. Two of them (Project and WorkDir) must be set in order to make *MOLCAS* work at all. There are defaults for these but you are adviced not to use the defaults.

There are several ways of using *MOLCAS* environment variables:

- These variables can be exported in your shell script

```
export MOLCASMEM=512
molcas input
```

- These variables can be included into *MOLCAS* input:

```
* begin of the input file
>>> export MOLCASMEM=512
  . . .
```

- variables can be included directly into molcas command in the form:

```
molcas MOLCASMEM=512 input
```

The simplest way to set up default environment for molcas is to use script ***`setuprc`***, which can be run as command molcas setuprc. This interactive script creates a resource file ***`molcasrc`***, located either in $\$MOLCAS$ or $\$HOME$ directory. The priority of these settings is: user defined settings (e.g. in molcas command), user resource file, molcas resource file.

Two flags in *MOLCAS* command are related to resource files:

Variable	*Purpose*
-env	Display current MOLCAS environment e.g. molcas -env input will print information about environment variables, used during execution of the input file.
-ign	Ignore resource files e.g. molcas -ign input will process input file without settings, which are stored in $\$MOLCAS/molcasrc$ and in $\$HOME/molcasrc$ files.

The complete list of MOLCAS-related environment variables:

Variable	*Purpose*
MOLCAS_WORKDIR	A parent directory for all scratch areas. It can be set to an absolute PATH (recommended), to a relative PATH, or to a spesial value PWD (to use current directory for scratch files)
MOLCAS_NEW_WORKDIR	If set to YES molcas will never reuse files in scratch area. This setting can be overwritten by running molcas with flag -old: molcas -old input

MOLCAS_KEEP_FILES if set to NO molcas will remove scratch area after a calculation. This setting can be overwritten by running molcas with flag -clean.

MOLCAS_PROJECT If set to value NAME, *MOLCAS* will use the prefics of the input file as a project name. Otherwise, it set a project name for the calculation. If set to the value NAMEPID, the Project name still will be constracted from the name of input file, however, the name of scratch area will be random.

MOLCAS_OUTPUT This variable can alter the default directory for extra output files, such as orbitals files, molden files, etc. If set, molcas will save output files to the specified directory. The directory name can be set in the form of absolute PATH, or relative PATH (related to the submit directory). A special value 'WORKDIR' will keep all output files in WorkDir. A special value 'NAME' will create a subdirectory with a name of Project. If the variable is not set, all output files will be copied or moved to the current directory.

MOLCAS_SAVE This variable can alter the default filenames for output files. If not set (default), all files will overwrite old files. If set to 'INCR' all output files will get an incremental filenames. If set to 'ORIG' - an existent file will be copied with an extension '.orig'

MOLCASMEM This environment variable controls the size of the work array utilized in the programs that offer dynamic memory. It is specified in Megabytes, i.e.
`MOLCASMEM=256; export MOLCASMEM`
will assign 256MB for the working arrays.

- MOLCASMEM is undefined — The default amount of memory (256MB), will be allocated for the work arrays.
- MOLCASMEM=0 (zero) — The largest possible work array that the system setup allows will be allocated, but at most 256MB.
- MOLCASMEM is defined but nonzero — This amount of memory will be allocated.

MOLCASDISK The value of this variable is used to split large files into a set of smaller datasets, as many as are needed (max. 20 subsets). It is specified in Megabytes, for instance, `MOLCASDISK=1000; export MOLCASDISK`, and the following rules apply:

- MOLCASDISK is undefined — The program modules will ignore this option and the file size limit will be defined by your hardware (2 GBytes for 32-bit machines).
- MOLCASDISK=0 (zero) — The programs will assume a file size limit of 2 GBytes (200GBytes on 64-bit machines).
- MOLCASDISK is defined but nonzero — The files will be limited to this value (approximately) in size.

MOLCASRAMD	Note: This variable is used only in `SEWARD` and is NOT recommended for other modules - on some systems using of this variable can decrease the memory, available for system I/O cashing, and so drastically decrease the performance. If your system is equipped with a large amount of memory you may avoid the I/O bottleneck by using that memory as a 'silicon' disk for the two-electron integrals in AO-basis. The process is controlled by the value of the variable MOLCASRAMD. It is specified in Megabytes and the following rules apply: • MOLCASRAMD is undefined or zero — The program modules will ignore this option. • MOLCASRAMD is defined but nonzero — At program start up time the two-electron integrals will be copied into memory. If there is not sufficient space available the program will resume normal activity.
MOLCAS_MOLDEN	If MOLCAS_MOLDEN set to 'ON' a Molden style input file will be generated regardless of the number of orbitals.
MOLCAS_PROPERTIES	If MOLCAS_PROPERTIES is set to 'LONG' properties with the individual MO contributions will be listed.
MOLCAS_TRAP	If MOLCAS_TRAP set to 'OFF' `AUTO` will continue execute molcas modules, even if non-zero return code was produced.
MOLCAS_PRINT	MOLCAS_PRINT variable controls the level of output. The value could be numerical or mnemonic: SILENT (0), TERSE (1), NORMAL (2), VERBOSE (3), DEBUG (4) and INSANE (5).
MOLCAS_REDUCE_PRT	If set to NO, print level in DO WHILE loop is not reduced.
MOLCAS_LINK	An environment variable to control information about linking of files. By default (MOLCAS_LINK is not set) only essential information about linking will be printed. To increase/decrease the print level, set MOLCAS_LINK to 'Yes'/'No'.
MOLCAS_LICENSE	An environment which specifies the directory with molcas lisence file *`license.dat`*. The default value of this variable is $MOLCAS.

7.3 General input structure. EMIL commands

This is a general guide to the input structure of the programs in the *MOLCAS* program system. All programs conform to the same conventions except where explicitly stated otherwise.

The programs are driven by keywords, which are either used without further information, or followed by additional specifications on the line(s) following the keyword, and is normally numeric in nature. *All numerical inputs are read in free format, note that in general MOLCAS will not be able to process lines longer than 120 characters.* The keywords can be given in mixed case (both upper and lower case are allowed). In the input stream you can insert comment lines anywhere, except between a keyword and the following additional specifications, with a comment line identified by an asterisk (*) in the first position on the line.

Most codes look at the first 4 characters of the keyword and ignores the rest. The entries in the lists of keywords below follow the standard that the significant characters are in upper case and larger than the nonsignificant characters. This do not imply that the keywords have to be typed in upper case; they can be typed freely in mixed case.

All inputs begin with a name of the program preceeding the keywords:

```
␣&PROGRAM
*␣here␣follows␣the␣keywords
```

where PROGRAM is the name of the MOLCAS module. The input listing is finished when a new program name, preceeded by the symbol &, is found (or the end of file or an EMIL command).

The following is an example of a list of keywords common to most of the programs:

Keyword	*Meaning*
TITLe	This keyword starts the reading of title line. The following line is treated as title line.

The programs only decode the first four characters of a keyword (except otherwise specifically indicated). For clarity it is however recommended to write the full keyword name. The keywords can be typed freely in upper, lower or mixed case.

An example for an input file used to run the `SCF` program follows:

```
␣&SCF
Title
␣Water␣molecule.␣Experimental␣equilibrium␣geometry
*␣The␣symmetries␣are:␣a1,␣b2,␣b1␣and␣a2.
Occupied
3␣1␣1␣0
*␣The␣ivo␣keyword␣prepares␣virtual␣orbitals␣for␣MCSCF.
Ivo
```

MOLCAS contains an input-oriented shell-script named `AUTO` that is implemented into the MOLCAS framework as if it was a program module. The shell will run the MOLCAS programs sequentially in the order they appear in the general input file. The script allows to orginaize loops (for structure optimization), and execute modules or commands conditionally.

In the first step of execution `AUTO` analyses user's input, cleans it (removes tabulation signs, DOS-style end-of-lines, etc.), and makes some preprocessing of the input.

The internal language used by `AUTO` is EMIL (Extended Molcas Input Language). It includes three different types of input commands:

- Sections with MOLCAS input.
- EMIL commands (a line started with > character)
- UNIX commands

7.3.1 Molcas input

EMIL allows to write MOLCAS input in a more compact way: user can omit &*END*, as well as a compulsory (in previous versions of molcas) keyword END OF INPUT. As soon as a new module (or UNIX, or EMIL command) is requested in a user input, the input for the module is terminated.

Also, it is possible to separate lines by ; sign, or by = sign (to create a pair *keyword* = *value*). In some rare occasions signs ; and = are used in the input for a molcas module. In order to keep these symbols unchanged, user can mark a part of an input, containing these symbols, by EMIL commands >> *verbatim* and >> *endverbatim*.

It means that the input:

```
 &SCF &END
CHARGE
 1
End of input
 &ALASKA &END
End of input
 &SLAPAF &END
End of input
```

could be written as:

```
 &SCF; CHARGE=1
 &ALASKA;  &SLAPAF
```

User can comment parts of input, by using $*$ at the beginning of line, or use C-style comments ($/*$ ... $*/$) to comment several lines.

Including in the input file a UNIX command preceded by an exclamation mark allows to execute the command during the execution of `AUTO`. For instance the listing command !ls -ls. One can use one or two exclamation marks, the difference being that with two the command will only be executed on the master node, i.e. in serial executions there's no difference between the two forms. Note, that UNIX command export is not allowed in `AUTO`. User should use EMIL command EXPORT instead.

7.3.2 EMIL commands

EMIL commands can be written in a short form:

```
> KEY [VALUE]
```

or in a nice form:

```
>>>>>>>>>>  KEY  [VALUE]  <<<<<<<<<
```

Here is a list of EMIL commands:

Command	*Purpose*
>> EXPORT A=B	a command to set environment variable A to value B

>> EXIT	a command to terminate execution. An optional value for this command is the return code (default value is 0)
>> INCLUDE file	a command to include a file into the input A compulsory value for this command is the filename.
>> UNIX	a command to execute a UNIX command. This EMIL command is an alias for '!'.
>> LINK	a command to make a link between two files. The command is similar to '!ln -s FILE1 FILE2' but can be used also in parallel environment. >>LINK -FORCE is a portable implementation of '!ln -fs FILE1 FILE2'.
>> COPY	a command to make a copy. The command is similar to '!cp -f FILE1 FILE2' but can be used also in parallel environment. Note that EMIL command does not allow to use masks in the command.
>> RM	a command to delete a file. The command is similar to '!rm -f FILE' but can be used also in parallel environment. Note that EMIL command does not allow to use masks in the command.
>> EVAL A=B	evaluate a numerical value

Keywords to organize loops in input, and execute modules conditionally:

Command	*Purpose*
>> DO WHILE	a command to start a loop. The loop should be terminated by SLAPAF or LOOP module, followed by ENDDO command
>> FOREACH A in (B, C, D)	a command to loop when the value of A is in the coma separated list. The list also can be written in the format "From .. To". Note that variable in the loop must be uppercased.
>> ENDDO	a command to finish the loop. If last module (before ENDDO command) returns 1 - the loop will be executed again (if number of iterations is less than MAXITER). If the return code is equal to 0 the loop will be terminated.
>> IF (ITER = N)	- a command to make conditional execution of modules/commands on iteration N (N possibly could be a space separated list)
>> IF (ITER != N)	- a command to skip execution of modules/commands on iteration N
>> IF ($VAR = N)	- a command to make conditional execution if $VAR value equals to N (if statement terminated by ENDIF command)
>> IF ($VAR = N) GOTO JUMP	- a command to make conditional goto to a label JUMP

>> LABEL JUMP	- a command to define a label.Note! Only forward jumps are allowed.
>> ENDIF	terminate IF block. Note nested if's are not allowed.

`AUTO` automatically stops calculation if a module returns a returncode higher than 0 or 1. To force `AUTO` to continue calculation even if a returncode equal to 16 (which is a return code for non-convergent calculation) one should set environment variable MOLCAS_TRAP='OFF'.

`SLAPAF` returns a special return code in the case of converged (non converged) geometry. So, to organize a structure calculation one should place the call to `SLAPAF` as a last statement of loop block. The summary of geometry optimization convergence located in a file *`$Project.structure`*. The programs following a geometry optimization will automatically assume the optimized geometry and wave function. Any new `SEWARD` calculation after an optimization (minimum or transition state) will disregard the input coordinates and will take the geometry optimized by `AUTO`.

It is also possible to use a special dummy program `LOOP` to organize infinite loops, or loops terminated by the counter (set be MOLCAS_MAXITER)

SET - an auto command to change settings of the script

Command	*Purpose*
>> SET MAXITER N <<	set maximum number of loop iterations to N. This option is obsolete and can be substituted by >> $exportMOLCAS\_ITER =$
>> SET OUTPUT SCREEN <<	redirect output (in loops) to screen. This option is obsolete.
>> SET OUTPUT FILE <<	redirect output (in loops) to a set of files in WorkDir. This option is obsolete.
>> SET OUTPUT OVER <<	skip output during structure loops, and print only last iteration. This option is obsolete.

Verbatim input.

If an input for a module must contain special symbols, such as ; or =, user can mark a corresponding part of the input by EMIL command $VERBATIM$

Command	*Purpose*
>> VERBATIM <<	start verbatim input
>> END VERBATIM <<	finish verbatim input

Below are different input examples.

The first example shows the procedure to perform first a `CASSCF` geometry optimization of the water molecule, then a numerical hessian calculation on the optimized geometry, and later to make a `CASPT2` calculation on the optimized geometry and wave function. Observe that the position of the `SLAPAF` inputs controls the data required for the optimizations.

```
*
*␣␣␣␣Start␣Structure␣calculation
*
>>EXPORT␣MOLCAS_ITER=50
␣&GATEWAY
coord
Water.xyz
BASIS␣=␣ANO-S
>>>>>>>>>>>>>␣Do␣while␣<<<<<<<<<<<<
␣&SEWARD
>>>>>>>>␣IF␣(␣ITER␣=␣1␣)␣<<<<<<<<<<
␣&SCF
>>>>>>>␣␣ENDIF␣<<<<<<<<<<<<<<<<<<<<
␣&RASSCF
Title
␣H2O␣ANO(321/21).
Nactel␣␣␣=␣6␣␣0␣␣0
Spin␣␣␣␣␣=␣1
Inactive␣=␣1␣␣0␣␣0␣␣0
Ras2␣␣␣␣␣=␣3␣␣2␣␣1␣␣0
␣&ALASKA;␣&SLAPAF
>>>>>>>>>>>>>␣␣ENDDO␣␣<<<<<<<<<<<<<<
␣&CASPT2
Maxit␣=␣20
Lroot␣=␣1
␣&GRID_IT
```

Another example demonstarte a possibility to use loops. SCF module will be called twice - first time with BLYP functional, second time with B3LYP functional.

```
*-------------------------------------------------------
␣&GATEWAY
coord
$MOLCAS/Test/input/C2H6.xyz
basis
ANO-S-VDZ
group
y␣xz
*-------------------------------------------------------
␣&SEWARD
Title
Ethane␣DFT␣test␣job
*-------------------------------------------------------
>>foreach␣DFT␣in␣(BLYP,␣B3LYP␣)
␣&SCF␣;␣KSDFT␣=␣$DFT
>>enddo
*-------------------------------------------------------
The␣next␣examples␣calculates␣HF␣energy␣for␣the␣several␣structures:
*␣modify␣coordinates␣in␣place
>>foreach␣DIST␣in␣(1.0,␣2.0,␣20.0)
␣&GATEWAY
Coord
2
hydrogen␣molecule
H␣0␣0␣0
H␣$DIST␣0␣0
BASIS=␣ANO-S-MB
```

```
GROUP=␣C1
␣&SEWARD
␣&SCF
UHF
SCRAMBLE=0.3
>>enddo
*␣incremental␣change␣of␣coordinates
>>export␣DIST=1.0
>>foreach␣L␣in␣(␣1␣..␣3␣)
>>eval␣DIST=$DIST+0.1
␣&GATEWAY
Coord
2
hydrogen␣molecule
H␣0␣0␣0
H␣$DIST␣0␣0
BASIS=␣ANO-S-MB
GROUP=␣C1
␣&SEWARD
␣&SCF
>>enddo
*␣different␣coordinate␣files
>>foreach␣COO␣in␣(␣000,␣001,␣002)
␣&GATEWAY
Coord␣=␣H2$COO.xyz
BASIS=␣ANO-S-MB
GROUP=␣C1
␣&SEWARD
␣&SCF
>>enddo
```

7.3.3 Use of shell parameters in input

The *MOLCAS* package allows the user to specify parts or variables in the the input file with shell variables, which subsequently are dynamically defined during execution time. **Note:** the shell variable names must be in upper case. Find below a simple example where a part of the H_2 potential curve is computed. First, the script used to run the calculation:

```
#! /bin/sh
#
Home=‘pwd‘ ;                     export Home
Project=H2 ;                     export Project
WorkDir=/tmp/$Project ;          export WorkDir
#
# Create workdir and cd to it
#
rm -fr $WorkDir
mkdir $WorkDir
#
# Loop over distances
#
for R in 0.5 0.6 0.7 0.8 0.9 1.0
do
   export R
   molcas $Home/$Project.input > $Home/$Project-$R-log 2> $Home/$Project-$R-err
done
#
# Cleanup WorkDir
#
rm -fr $WorkDir
```

In this sh shell script we have arranged the call to the *MOLCAS* package inside a loop over the various values of the distances. This value is held by the variable $R which is exported every iterations. Below is the input file used, note that the third cartesian coordinate is the variable $R.

```
␣&SEWARD
Symmetry
␣x␣y␣z
Basis␣set
H.sto-3g....
H␣␣␣0.000␣␣␣0.000␣␣␣$R
End␣of␣basis
End␣of␣input
␣&SCF
```

7.4 MOLCAS-7 Flowchart

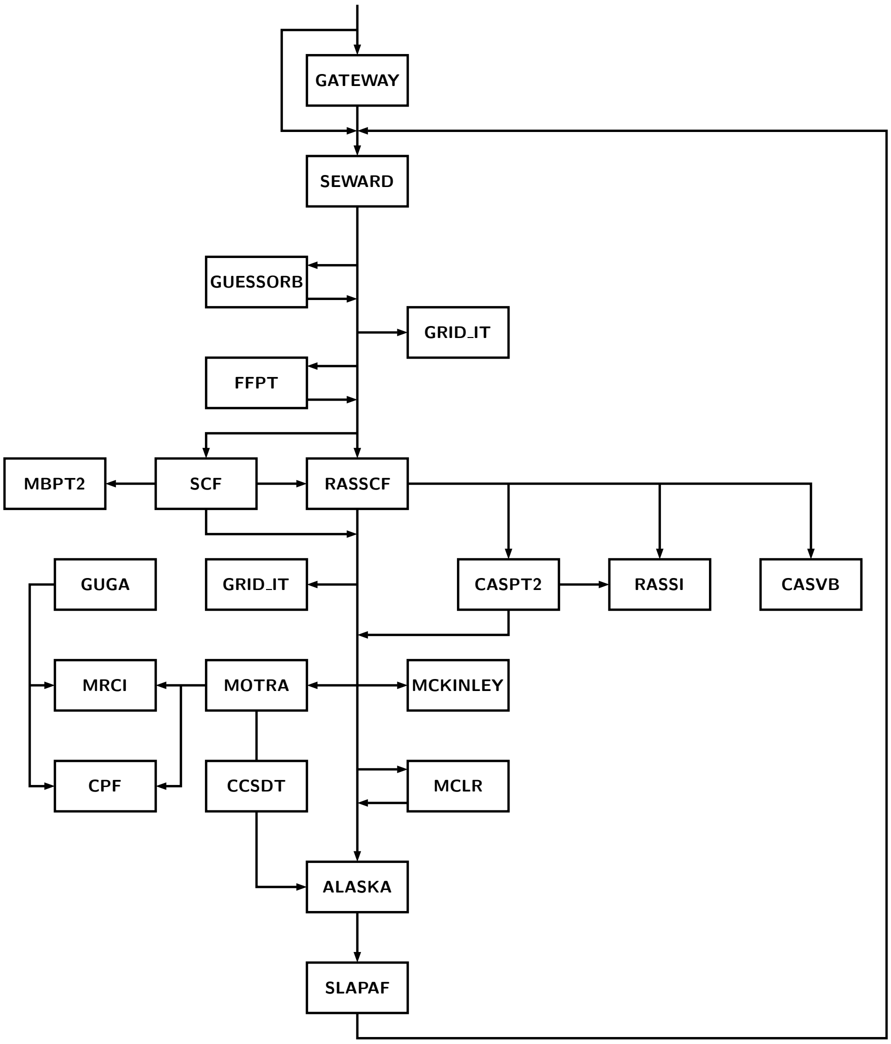


Figure 7.1: Program module dependencies flowchart for MOLCAS.

Section 8

Programs

8.1 ALASKA

This module is automatically invoked by the SLAPAF module. This is the preferred mode of operation! In connection with numerical gradients it will ensure that the rotational and translational invariance is fully utilized in order to reduce the number of used displacements.

The ALASKA module compute analytic or for numerical gradients requests the execution of an alternative module. The ALASKA module figures out the method automatically. Analytic methods are implemented for the HF, KS-DFT, and RASSCF and SA-CASSCF method. Numerical methods are implemented for SCF, KS-DFT, RASSCF, MBPT2, CCSDT, the CASPT2 and MS-CASPT2 methods, including the use of the Cholesky decomposition for the methods were that has been implemented.

Both analytic and numerical procedures are parallelized.

For SA-CASSCF gradient the ALASKA module will automatically start up the MCLR module if required.

8.1.1 Analytic Gradients

Gradients of the energy with respect to nuclear coordinates can be computed for any type of wave function as long as an effective first order density matrix, an effective Fock matrix, and an effective second order density matrix is provided. The term effective is related to that these matrices in the case of non-variational parameters in the wave function (e.q. CI, MP2, CASPT2, etc.) are modified to include contributions from the associated Lagrange multipliers. The gradient expression apart from these modifications is the same for any wave function type. ALASKA is the gradient program, which will generate the necessary integral derivatives and combine them with the matrices mentioned in the text above.

8.1.2 Description

ALASKA is written such that gradients can be computed for any kind of basis function that SEWARD will accept.

ALASKA is able to compute the following integral derivatives:

- overlap integrals,
- kinetic energy integrals,
- nuclear attraction integrals (point charges or finite nuclei),
- electron repulsion integrals,
- external electric field integrals,
- ECP and PP integrals,
- reaction field integrals,
- and Pauli repulsion integrals.

`ALASKA` employs two different integration schemes to generate the one- and two-electron integral derivatives. The nuclear attraction and electron repulsion integrals are evaluated by a modified Rys-Gauss quadrature [82]. All other integral derivatives are evaluated with the Hermite-Gauss quadrature. The same restriction of the basis sets applies as to `SEWARD`. None of the integral derivatives are written to disk but rather combined immediately with the corresponding matrix from the wave function.

At present the following limitations are built into `ALASKA`:

Max number of unique basis functions:	2000
Max number of symmetry independent centers:	500
Highest angular momentum:	15
Highest symmetry point group:	D_{2h}

8.1.3 Numerical gradients

The module is parallelized over the displacements, which in case of large jobs gives a linear speed up compared to a serial execution, although in order to obtain this it is important to choose the number of nodes such that the number of contributing perturbations is a multiple of the number of nodes. For a given molecule the number of perturbations equals the number of atoms times 6 (a perturbation with plus and minus delta for each of the three axises). Symmetry can of course reduce this number.

8.1.4 Dependencies

`ALASKA` depends on the density and Fock matrices generated by `SCF` or `RASSCF`. In addition it needs the basis set specification defined in `SEWARD`. The dependencies of the numerical part of the module is the union of the dependencies of the `SEWARD`, `SCF`, `RASSCF`, `MBPT2`, `MOTRA`, `CCSDT`, and `CASPT2` modules. All these dependencies, however, are totally transparent to the user.

8.1.5 Files

Input files

Apart from the standard input unit `ALASKA` will use the following input files.

File	*Contents*
RYSRW	Data base for the fast direct evaluation of roots and weights of the Rys polynomials. This file is a part of the program system and should not be manipulated by the user.
ABDATA	Data base for the evaluation of roots and weights of high order Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
ONEINT	One-electron integrals and auxiliary information.
RUNFILE	Auxiliary data generated by `SEWARD` and `GATEWAY`.

The files of the `SEWARD`, `SCF`, `RASSCF`, `MBPT2`, `MOTRA`, `CCSDT`, and `CASPT2` modules are needed for the numerical procedure.

Output files

In addition to the standard output unit `ALASKA` will generate the following files.

File	*Contents*
RUNFILE	The ***runfile*** is updated with information needed by the `SLAPAF` relaxation program. `ALASKA` will write the molecular Cartesian gradients on this file.
ALASKA.INPUT	File with the latest input processed by `ALASKA`.

8.1.6 Input

Below follows a description of the input to `ALASKA`. Note that input options are related to the analytic gradient procedure if not otherwise noted!

In addition to the keywords and the comment lines the input may contain blank lines. The input is always preceded by the program name:

```
␣&ALASKA
```

The first four characters of the keywords are decoded while the rest are ignored. However, for a more transparent input we recommend the user to use the full keywords.

Optional keywords for analytical gradients

Keyword	*Meaning*
ONEOnly	Compute only the nuclear repulsion and one-electron integrals contribution to the gradient. The default is to compute all contributions to the molecular gradient.
CUTOff	Threshold for ignoring contributions to the molecular gradient follows on the next line. The default is 1.0d-7. The prescreening is based on the 2nd order density matrix and the radial overlap contribution to the integral derivatives.
TEST	With this keyword the program will process only the input. It is a debugging aid to help you check your input.
POLD	The gradient is printed in the old format. Note: by default gradient is not printed any longer.
PNEW	The gradient is printed in the new human-readable format.
VERBose	The output will be a bit more verbose.
SHOW gradient contributions	The gradient contributions will be printed.

Optional keywords for numerical gradients

Keyword	*Meaning*
NUMErical	Forces the use of numerical gradients even if analytical ones are implemented. The default is to use analytical gradients whenever possible.
ROOT	For use with numerical gradients only! Specifies which root to optimize the geometry for, if there is more than one root to choose from. In a RASSCF optimization, the default is to optimize for the same root as is relaxed. In a MS-CASPT2 calculation, the default is to optimize for root 1.
DELTa	For use with numerical gradients only! The displacement for a given center is chosen as the distance to the nearest neighbor, scaled by a factor. This factor can be set through the DELTa keyword. The default value is 0.005.

The following is an example of an input which will work for almost all practical cases. Note that it is very rarely that you need to run this program explicitly. It is usually controlled by the program `SLAPAF`.

```
␣&Alaska
```

8.2 `CASPT2`

Second order multiconfigurational perturbation theory is used in the program `CASPT2` to compute the (dynamic) correlation energy [49, 50]. The zeroth order wave function is of the CAS type. The first step is therefore a CASSCF calculation and the CASPT2 calculation gives a second order estimate of the difference between the CASSCF and the full CI energy. The CASPT2 method has been tested in a large number of applications [42, 43]. Here follows a brief summary of results.

Bond distances are normally obtained with an accuracy of better that 0.01 Å for bonds between first and second row atoms. With the standard Fock matrix formulation, bond energies are normally underestimated with between 2 and 5 kcal/mol for each bond formed. This is due to a systematic error in the method[83]. In every process where the number of paired electrons is changed, an error of this size will occur for each electron pair. For example, the singlet-triplet energy difference in the methylene radical (CH_2) is overestimated with about 3 kcal/mol [50]. Heats of reactions for isogyric reactions are predicted with an accuracy of ±2 kcal/mol. These results have been obtained with saturated basis sets and all valence electrons active. The use of smaller basis sets and other types of active spaces may, of course, affect the error.

These systematic errors have recently been considerably reduced by the introduction of a modified zeroth order Hamiltonian [84]. The method introduces a shift (the IPEA shift) that modifies the energies of active orbitals such that they become closer to ionization energies when excited from and closer to electron affinities when excited out of. The approach has been tested for 49 diatomic molecules, reducing the mean error in D_0 from 0.2 to 0.1 eV. For the triply bonded molecules N_2, P_2, and As_2 it was reduced from 0.45 eV to less than 0.15 eV. Similar improvements were obtained for excitation and ionization energies. The IPEA modified H_0 (with a shift parameter of 0.25) is default in MOLCAS from version 6.4.

An alternative to IPEA is to use the options, called 'g_1', 'g_2', and 'g_3'(See Ref. [62]), that stabilizes the energies of the active orbitals. The remaining error is no longer systematic, and is generally reduced. For example, the error in the singlet-triplet separation of CH_2 is reduced to 1 kcal/mol [62]. This option is, however, not recommended any longer because it has been replaced by the IPEA Hamiltonian.

The CASPT2 method can be used in any case where a valid reference function can be obtained with the CASSCF method. There is thus no restriction in the number of open shells or the spin coupling of the electrons. Excited states can be treated at the same level as ground states. Actually one of the major successes with the method has been in the calculation of excitation energies. A large number of applications have been performed for conjugated organic molecules. Both Rydberg and valence excited states can be treated and the error in computed excitation energies is normally in the range 0.0–0.2 eV. Similar results have been obtained for ligand field and charge-transfer excitations in transition metal compounds. From MOLCAS-6 it is possible to use the CASPT2 method in conjunction with the Douglas-Kroll-Hess relativistic Hamiltonian, which has made possible calculations on heavy element compounds such a third row transition metal compounds and actinides with accurate results.

The CASPT2 method can also be used in combination with the `FFPT` program to compute dynamic correlation contributions to properties with good results in most cases. Numerical gradients are available with the `SLAPAF` module.

The CASPT2 method is based on second order perturbation theory. To be successful, the perturbation should be small. A correct selection of the active space in the preceding CASSCF calculation is therefore of utmost importance. All near-degeneracy effects leading to configurations with large weights must be included at this stage of the calculation. If this is not done, the first order wave function will contain large coefficients. When this occurs, the `CASPT2` program issues a warning. If the energy contribution from such a configuration is large, the results is not to be trusted and a new selection of the active space should be made.

Especially in calculations on excited states, intruder states may occur in the first order wave function. Warnings are then issued by the program that an energy denominator is small or negative. Such intruder states often arise from Rydberg orbitals, which have not been included in the active space. Even if this sometimes leads to large first order CI coefficients, the contribution to the second order energy is usually very small, since the interaction with the intruding Rydberg state is small. It might then be safe to neglect the warning. A safer procedure is to include the Rydberg orbital into the active space. It can sometimes be deleted from the MO space.

Calculations on compounds with heavy atoms (transition metals, actinides, etc) may yield many virtual orbitals with low energies. The interaction energies for excitations to states where these orbitals are occupied are often very small and the low denominators can then be removed by a suitable level shift (see below). But it is always safer to include such orbitals in the active space.

Two keywords have been introduced to deal with this fairly common situation, for excited states, that weakly coupled intruders cause spurious singularities, 'spikes' in e.g. a potential curve. The two keywords SHIFT and IMAGINARY SHIFT (mutually exclusive) will introduce a shift in the energy denominators, thus avoiding singularities, and will also correct the energy for the use of this shift. The net effect is that the energy is almost unaffected except in the vicinity of the weak singularity, which is removed. The SHIFT keyword adds a real shift, and the use of this procedure is well tested [52, 54]. The IMAGINARY SHIFT adds an imaginary quantity, and then uses the real value of the resulting second-order energy [53]. This offers some advantage, in particular for weak intruder states.

In some cases, where one can expect strong interaction between different CASSCF wave functions, it is advisable to use the Multi-State (MS) CASPT2 method [13]. A second order effective Hamiltonian is constructed for a number of CASSCF wave functions obtained in a state-average calculation. This introduces interaction matrix elements at second order between the different CASSCF states. The effective Hamiltonian is diagonalized to obtain the final second order energies. The program also produces a file (JOBMIX) with the new effective zeroth order wave functions, which are linear combinations of the original CASSCF states. This method has been used successfully to separate artificially mixed valence and Rydberg states and for transition metal compounds with low lying excited states of the same symmetry as the ground state.

It is clear from the discussion above that it is not a 'black box' procedure to perform CASPT2 calculations on excited states. It is often necessary to iterate the procedure with modifications of the active space and the selection of roots in the CASSCF calculation until a stable result is obtained. Normally, the CASSCF calculations are performed as average calculations over the number of electronic states of interest, or a larger number of states. It is imperative that the result is checked before the CASPT2 calculations are performed. The solutions should contain the interesting states. If all of them are not there, the number of roots in the

CASSCF calculation has to be increased. Suppose for example, that four states of a given symmetry are required. Two of them are valence excited states and two are Rydberg states. A CASSCF calculation is performed as an average over four roots. Inspection of the solution shows only one valence excited state, the other three are Rydberg states. After several trials it turns out that the second valence excited state occurs as root number seven in the CASSCF calculation. The reason for such a behavior is, of course, the very different dynamic correlation energies of the valence excited states as compared to the Rydberg states. It is important that the AO basis set is chosen to contain a good representation of the Rydberg orbitals, in order to separate them from the valence excited states. For more details on how to perform calculations on excited states we refer to the literature [52, 54] and section 5.5 of the examples manual.

The first order wave function is obtained in the `CASPT2` program as an iterative solution to a large set of linear equations. The size of the equation system is approximately $n^2 * m^2/2$ where n is the sum of inactive and active orbitals and m is the sum of active and secondary orbitals. Symmetry will reduce the size with approximately a factor g_{sym}, the number of irreps of the point group.

`CASPT2` produces a set of molecular orbitals that can be used as start orbitals for other programs or further calculations. A minimal CASSCF and CASPT2 gives orbitals and occupation numbers which can be used to design a proper larger calculation. By default, the orbitals are natural orbitals obtained from the density matrix of the (normalized) wave function through first order. However, the active/active block of that density matrix is not computed exactly. An approximation has been designed in such a way that the trace is correct, and the natural occupation numbers of active orbitals are between zero and two. Due to the approximation, any properties computed using these orbitals are inexact and can be used only qualitatively. An exact first order density matrix can be computed but this is more time-consuming. It is controlled by the keyword DENSity. Use this keyword to compute properties like dipole moments, etc. The most secure accurate way to do that is. however, to use finite field perturbation theory (FFPT).

For compatibility with earlier programs, two keywords are available that change the default definition of the output orbitals. Using the keyword MOLOrb, you will obtain orbitals that are identical to the natural orbitals from the `RASSCF` calculation in the inactive and active subspaces, while the secondary orbitals are obtained by diagonalizing the secondary subspace of the density matrix of the (normalized) perturbed wave function. This is often useful for preparing orbital sets for subsequent calculations. The `RASSCF` calculation can be reproduced with any or several virtual orbitals deleted. Therefore, the virtual space can be trimmed by deleting orbitals with low occupation number. Also, an intruder due to a deficient active space will produce a virtual orbital with large occupation number. Inclusion of this orbital into the active space eliminates the intruder. Similarly, if the intruder is of the weak ‘accidental’ type, that orbital can be deleted.

Using the NATUral keyword, you will get the natural orbitals obtained from the density matrix through first order, either in the approximate form (default) or in the exact form by the use of the keyword DENSity.

Requirements: In the `RASSCF` calculation RAS1 and RAS3 have to be empty. (Pure CASSCF calculations only, see program and input descriptions for `RASSCF`.) It is possible to have an empty RAS2 space (closed shell). However, if only the second order energy is required it is in this case faster to first make an `SCF` calculation and then an `MBPT2` calculation.

8.2.1 Dependencies

The `CASPT2` program needs the JOBIPH file from a `RASSCF` calculation, and in addition one- and two-electron integrals and some auxiliary files from `SEWARD`.

8.2.2 Files

Like all the MOLCAS programs, CASPT2 opens the RUNFILE file.

Input files

File	*Contents*
*ORDINT**	Two-electron integrals from `SEWARD`. Actually, a multifile system, using several files named ORDINT, ORDINT1,. . . .
ONEINT	One-electron integrals from `SEWARD`.
JOBIPH	A `RASSCF` interface file.
RUNFILE	A file with auxiliary data produced by `GATEWAY` and`SEWARD`.

Output files

File	*Contents*
PT2ORB	Molecular orbitals.

8.2.3 Input

This section describes the input to the `CASPT2` program, starting with its name:

```
 &CASPT2
```

Keywords

Keyword	*Meaning*
TITLE	Following this keyword, enter one title line for this job.

MULTistate Enter number of root states, and a list of which CI vector from the CASSCF calculation to use for each state. Also used for single-state calculations, when the root state is not the ground state. (The keyword LROOT is obsolete and should not be used.)

IPEAshift The shift, given in au. and weighted by a function of the diagonal density matrix elements for the active orbitals, D_{pp}, corrects the energies of the active orbitals. This option is used to modify the standard definition of the zeroth order Hamiltonian (H_0), which includes an IPEA shift of 0.25 [84]. The modification of H_0 has been introduced (Nov 2005) to reduce the systematic error which leads to a relative overestimation of the correlation energy for open shell system. It also reduces the intruder problems. Use this keyword only for testing purposes. Default is to use an IPEA shift of 0.25.

IMAGinary Add an imaginary shift to the external part of the zero order Hamiltonian. The correlation energy computed is the real part of the resulting complex perturbation energy. Also, a corrected value, obtained by Hylleraas' variational formula, is computed. See Ref. [53]. As with the real shift, this option is used to eliminate intruder problems.

SHIFt Add a shift to the external part of the zero order Hamiltonian. See Refs. [53, 52, 43]. In addition to the conventionally computed second order energy value, another energy obtained by Hylleraas' variational formula is computed. This energy is then very close to the unshifted energy, except close to singularities due to intruders. This option should only be used to eliminate intruder state problems.

AFRE This keyword is used to select atoms for defining the correlation orbital space for the CASPT2 calculation. Assume that you have a large molecule where the activity takes place in a limited region (the active site). It could be a metal atom with its surrounding ligands. You can then use this option to reduce the size of the CASPT2 calculation by freezing and deleting orbitals that have only a small population in the active site. An example: The cobalt imido complex Co^{III}(nacnac)(NPh) has 43 atoms. The active site was cobalt and the surrounding ligand atoms. Using the AFRE option reduces the time for the CASPT2 calculation from 3 hrs to 3 min with a loss of accuracy in relative energies for 24 electronic states of less than 0.1 eV. The first line after the keyword contains the number of selected atoms then the selection thresholds (the recommended value is 0.1 or less). An additional line gives the names of the atoms as defined in the Seward input. Here is a sample input for the cobalt complex mentioned above.

```
AFRE
␣6␣0.10␣0.00
␣Co␣N1␣N2␣C5␣C6␣C7
```

This input means that inactive orbitals with less than 0.1 of the density on the active sites will be frozen, while no virtual orbitals will be deleted.

LOVCaspt2 “Freeze-and-Delete” type of CASPT2, available only in connection with Cholesky or RI. An example of input for the keyword LOVC is the following:

```
LovCASPT2
␣0.3
DoMP2
```

In this case, both occupied and virtual orbitals (localized by the program) are divided in two groups: those mainly located on the region determined (automatically) by the spatial extent of the active orbitals (“active site”), and the remaining ones, which are obviously “outside” this region. The value of the threshold (between 0 and 1) is used to perform this selection (in the example, 30% of the gross Mulliken population of a given orbital on the active site). By default, the CASPT2 calculation is performed only for the correlating orbitals associated with the active site. The keyword DoMP2 is optional and forces the program to perform also an MP2 calculation on the “frozen region”. Alternatively, one can specify the keyword VirAll in order to use all virtual orbitals as correlating space for the occupied orbitals of the active site.

FNOCaspt2 Performs a Frozen Natural Orbital (FNO) CASPT2 calculation, available only in combination with Cholesky or RI integral representation. Needs (pseudo)canonical orbitals from RASSCF. An example of input for the keyword FNOC is the following:

```
FNOCaspt2
0.4
DoMP2
```

The keyword FNOC has one compulsory argument (real number in]0,1]) specifying the fraction of virtual orbitals (in each irrep) to be retained in the FNO-CASPT2 calculation. The keyword DoMP2 is optional and used to compute the (estimated) correction for the truncation error.

FOCKtype The line starts with “FOCK...” and ends with “...=XX”, or it has “FOCK...” as a keyword on one line, and the value “XX” on the next. Here, XX is the name of any of a number of possible variant Fock matrix definitions. The default is the simple Fock matrix described in [49, 50] and the other original CASPT2 references. The only official variants in MOLCAS are “G1”, “G2”, and “G3”. These refer to the three modifications described in ref. [62]. Note: from 6.4 it is not recommended to use this keyword but stay with the IPEA modified H_0, which is default.

FROZen This keyword is used to specify the number of frozen orbitals, i.e. the orbitals that are not correlated in the calculation. The next line contain the number of frozen orbitals per symmetry. The default is to freeze the max of those that were frozen in the `RASSCF` calculation and the deep core orbitals. The frozen orbitals are always the first ones in each symmetry.

DELEted	This keyword is used to specify the number of deleted orbitals, i.e. the orbitals that are not used as correlating orbitals in the calculation. The next line contain the number deleted orbitals per symmetry. The default is to delete those that were deleted in the `RASSCF` calculation. The deleted orbitals are always the last ones in each symmetry.
DENSity	Computes the full density matrix from the first order wave function, rather than approximated as is the (faster) default option. Used to compute CASPT2 properties, such as dipole moments, etc.
RFPErt	This keyword makes the program add reaction field effects to the energy calculation. This is done by adding the reaction field effects to the one-electron Hamiltonian as a constant perturbation, i.e. the reaction field effect is not treated self consistently.
RLXRoot	Specifies which root to be relaxed in a geometry optimization of a multi state CASPT2 wave function. Defaults to the highest root or root defined by the same keyword in the RASSCF module.
LROOt	Obsolete keyword. See keyword MULTistate.
THREsholds	On next line, enter two thresholds: for removal of zero-norm components in the first-order perturbed wave function, and for removal of near linear dependencies in the first-order perturbed wave function. Default values are 1.0d-10 and 1.0d-08 respectively.
MAXIter	On next line, enter the maximum allowed number of iterations in a procedure for solving a system of linear equations using a conjugate gradient method. Default is 20. A gradient norm is reported. This gradient is a residual error from the CASPT2 equation solution and should be small, else the number of iterations must be increased.
CONVergence	On next line, enter the convergence threshold for the procedure described above. The iterative procedure is repeated until the norm of the residual (RNORM) is less than this convergence threshold. Default is 1.0d-06.
MOLOrb	This keyword gives backwards compatibility to earlier CASPT2 programs. It specifies that a set of output orbitals will be created, which is identical to the CASSCF orbitals, except that the virtual orbitals are the natural orbitals of the (normalized) virtual/virtual part of the density matrix of the perturbed wave function.
NATUral	This keyword gives backwards compatibility to earlier CASPT2 programs. It specifies that a set of output orbitals will be created, that are the natural orbitals of the first-order density matrix. Note that it is necessary to use the keyword DENSity to obtain an exact density matrix. Otherwise, only an approximate density matrix is created.

NOMIX	Normally, a Multistate CASPT2 calculation produces new jobiph file named JOBMIX. It has the same CASSCF wave functions as the original ones, except that those CI vectors that was used in the Multistate CASPT2 calculation have been mixed, using the eigenvectors of the effective Hamiltonian matrix as transformation coefficients. Keyword NOMIX prevents creation of this JOBMIX file.
NOMUlt	This keyword removes the multistate part of the calculation and only runs a series of independent CASPT2 calculations for the roots specified by the MULTistate keyword. Useful when many roots are required, but multistate is not needed, or desired. Note that a JOBMIX file is produced anyway, but the vectors will not be mixed, and the energies will be single-state CASPT2 energies.
NOORbitals	In calculations with very many orbitals, use this keyword to skip the printing of the MO orbitals.
NOPRop	Normally, a CASPT2 calculation produces an exact or approximate density matrix, natural orbitals, and properties. Keyword NOPROP inhibits these calculations, saving time and memory.
NOTRansform	This keyword specifies that the wave function should not be transformed to use quasi-canonical orbitals, even if CASPT2 does not know if this was done or not and by default would do such a transformation. Effectively, the Fock matrix is replaced by a diagonal approximation in the input orbital system.
TRANsform	This keyword specifies that the wave function should be transformed to use pseudo-canonical orbitals, even if this was specified as option to the CASSCF calculation and should be unnecessary. (Default is: to transform when necessary, and not else.)
OUTPut	Use this keyword, followed by any of the words BRIEF, DEFAULT, or LONG, to control the extent of orbital listing. BRIEF gives a very short orbital listing, DEFAULT a normal output, and LONG a detailed listing. The keyword and value can also be given in a single line, in the form e.g. OUTPUT=BRIEF.
PACK	Threshold for packing some files to be written into disk. Accepted for back compatibility of input – this input does nothing nowadays.
PRWF	This keyword is used to specify the threshold for printing the CI coefficients. The value is specified on the next line, and the default is 0.05.

The given default values for the keywords Convergence and Thresholds normally give a second order energy which is correct in eight decimal places.

Input example

```
␣&CASPT2
```

```
Title
␣The␣water␣molecule
MaxIter
␣20
frozen
␣1␣0␣0␣0
Density␣matrix
```

The CASPT2 energy and density matrix is computed for the water molecule with the O(1s) orbital frozen. The standard IPEA-H_0 is used.

8.3 CASVB

This program can be used in two basic modes:

a) variational optimization of quite general types of nonorthogonal MCSCF or modern valence bond wavefunctions

b) representation of CASSCF wavefunctions in modern valence form, using overlap- (*relatively inexpensive*) or energy-based criteria.

For generating representations of CASSCF wavefunctions, the program is invoked by the command `CASVB`. For variational optimization of wavefunctions it is normally invoked inside `RASSCF` by the sub-command `VB` (see 8.30.3).

Bibliography: see [85, 86, 87, 88].

8.3.1 Dependencies

The `CASVB` program needs the JOBIPH file from a `RASSCF` calculation, and in addition also the ONEINT and ORDINT files from `SEWARD`.

8.3.2 Files

Like all the MOLCAS programs, CASVB opens the RUNFILE.

Input files

File	*Contents*
*ORDINT**	Two-electron integrals from `SEWARD`. Actually, a multifile system, using ten files named ORDINT, ORDINT1,… ORDINT9.
ONEINT	One-electron integrals from `SEWARD`.
JOBIPH	A `RASSCF` interface file.
VBWFN	Valence bond wavefunction information (orbital and structure coefficients). Typically this file is obtained from a previous invokation of the `CASVB` program.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

Output files

File	*Contents*
JOBIPH	On exit, the `RASSCF` interface file is overwritten with the CASVB wavefunction.
VBWFN	Valence bond wavefunction information (orbital and structure coefficients).

8.3.3 Input

This section describes the input to the `CASVB` program. The input for each module is preceded by its name like:

```
␣&CASVB
```

Keywords

Compulsory keywords

Keyword	*Meaning*
END OF INPUT	This marks the end of the input to the program.

Optional keywords to define the CASSCF wavefunction. Not generally required because values stored in the job interface file or used by the `RASSCF` program will normally be appropriate.

Keyword	*Meaning*
FROZEN	Specifies frozen orbitals, as in the `RASSCF` program.
INACTIVE	Specifies inactive orbitals, as in the `RASSCF` program.
NACTEL	Specifies the number of active electrons, as in the `RASSCF` program.
RAS2	Specifies RAS2 orbitals, as in the `RASSCF` program.
SPIN	Specifies the total spin, as in the `RASSCF` program.

SYMMetry	Specifies the CASSCF wavefunction symmetry, as in the `RASSCF` program.

Optional keywords to define the VB wavefunction

Keyword	*Meaning*
CON	The spatial VB configurations are defined in terms of the active orbitals, and may be specified using one or more `CON` keywords: `CON` $n_1\ n_2\ n_3\ n_4\ \ldots$ The configurations can be specified by occupation numbers, so that n_i is the occupation of the ith valence bond orbital. Alternatively a list of $Nact$ orbital numbers (in any order) may be provided – the program determines which definition applies. The two specifications `1 0 1 2` and `1 3 4 4` are thus equivalent. Input configurations are reordered by `CASVB`, so that configurations have non-decreasing double occupancies. Configurations that are inconsistent with the value for the total spin are ignored. If no configurations are specified the single 'covalent' configuration $\phi_1\phi_2\cdots\phi_{Nact}$ is assumed.
COUPle	`COUPLE` *key* *key* may be chosen from `KOTANI` (default), `RUMER`, `PROJECT` or `LTRUMER`, specifying the scheme for constructing the spin eigenfunctions used in the definition of valence bond structures. `PROJECT` refers to spin functions generated using a spin projection operator, `LTRUMER` to Rumer functions with the so-called "leading term" phase convention.
WAVE	`WAVE` $N\ S_1\ S_2\ \ldots$ This keyword can be used to specify explicitly the number of electrons and spin(s) to be used with a configuration list. If N is less than the present number of active electrons, the input wavefunction fragment is assumed to form part of a direct product. Otherwise, the spins specified may be greater than or equal to the SPIN value specified as input to the `RASSCF` program. Defaults, for both N and S, are the values used by `RASSCF`.

Optional keywords for the recovery and/or storage of orbitals and vectors

Keyword	*Meaning*
STARt	`START` *key*-1=*filename*-1 *key*-2=*filename*-2

	... Specifies input files for VB wavefunction (*key-i* = VB), CASSCF CI vector (*key-i* = CI) and/or CASSCF molecular orbitals (*key-i* = MO). By default, the required information is taken from the file JOBOLD.
SAVE	`SAVE` *key*-1=*filename*-1 *key*-2=*filename*-2 ... Specifies output files for VB wavefunction (*key-i* = VB) and/or the VB CI vector (*key-i* = VBCI). By default, the VB CI vector is written to the file JOBIPH.

Optional keywords to override the starting guess

Keyword	*Meaning*

GUESs	`GUESS`
	key-1 ...
	key-2 ...
	`ENDGUESs`
	
	The `GUESS` keyword initiates the input of a guess for the valence bond orbitals and/or structure coefficients. *key-i* can be either `ORB` or `STRUC`. These keywords modify the guess provided by the program. It is thus possible to modify individual orbitals in a previous solution so as to construct the starting guess. The `ENDGUESs` keyword terminates the guess input.
	
	`ORB`
	$i\ c_1\ c_2\ \ldots c_{mact}$
	Specifies a starting guess for valence bond orbital number i. The guess is specified in terms of the *mact* active MOs defining the CASSCF wavefunction.
	
	`STRUC`
	$c_1\ c_2\ \ldots c_{NVB}$
	Specifies a starting guess for the NVB structure coefficients. If this keyword is not provided, the perfect-pairing mode of spin coupling is assumed for the spatial configuration having the least number of doubly occupied orbitals. Note that the definition of structures depends on the value of `COUPLE`. Doubly occupied orbitals occur first in all configurations, and the spin eigenfunctions are based on the singly occupied orbitals being in ascending order.

ORBPerm	`ORBPERM` $i_1\ \ldots i_{mact}$

	Permutes the orbitals in the valence bond wavefunction and changes their phases according to $\phi'_j = \mathrm{sign}(i_j)\phi_{\mathrm{abs}(i_j)}$. The guess may be further modified using the `GUESS` keyword. Additionally, the structure coefficients will be transformed according to the given permutation (note that the configuration list must be closed under the orbital permutation for this to be possible).

Optional keywords for optimization control

Keyword	*Meaning*
CRIT	`CRIT` *method* Specifies the criterion for the optimization. *method* can be `OVERLAP` or `ENERGY` (`OVERLAP` is default). The former maximizes the normalized overlap with the CASSCF wavefunction: $\max\left(\frac{\langle\Psi_{CAS}\|\Psi_{VB}\rangle}{(\langle\Psi_{VB}\|\Psi_{VB}\rangle)^{1/2}}\right)$ and the latter simply minimizes the energy: $\min\left(\frac{\langle\Psi_{VB}\|\hat{H}\|\Psi_{VB}\rangle}{\langle\Psi_{VB}\|\Psi_{VB}\rangle}\right).$
MAXIter	`MAXITER` N_{iter} Specifies the maximum number of iterations in the second-order optimizations. Default is N_{iter}=50.
(NO)CASProj	`(NO)CASPROJ` With this keyword the structure coefficients are picked from the transformed CASSCF CI vector, leaving only the orbital variational parameters. For further details see the bibliography. This option may be useful to aid convergence.
SADDle	`SADDLE` n Defines optimization onto an n^{th}-order saddle point. See also T. Thorsteinsson and D. L. Cooper, Int. J. Quant. Chem. **70**, 637–50 (1998).
(NO)INIT	`(NO)INIT` Requests a sequence of preliminary optimizations which aim to minimize the computational cost while maximizing the likelihood of stable convergence. This feature is the default if no wavefunction guess is available and no `OPTIM` keyword specified in the input.

METHod

`METHOD`
key

Selects the optimization algorithm to be used. *key* can be one of: `FLETCHER`, `TRIM`, `TRUSTOPT`, `DAVIDSON`, `STEEP`, `VB2CAS`, `AUGHESS`, `AUG2`, `CHECK`, `DFLETCH`, `NONE`, or `SUPER`. Recommended are the direct procedures `DFLETCH` or `AUGHESS`. For general saddle-point optimization `TRIM` is used. Linear (CI only) optimization problems use `DAVIDSON`. `NONE` suspends optimization, while `CHECK` carries out a finite-difference check of the gradient and Hessian.

The default algorithm chosen by `CASVB` will be usually be adequate.

TUNE

`TUNE`
...

Enables the input of individual parameters to be used in the optimization procedure (*e.g.* for controlling step-size selection and convergence testing). Details of the values used are output if *print*(3)$\geq$3 is specified. For expert use only.

OPTIm

More than one optimization may be performed in the same `CASVB` run, by the use of `OPTIM` keywords:

`OPTIM`
[...
`ENDOPTIM`]

The subcommands may be any optimization declarations defined in this section, as well as any symmetry or constraints specifications. Commands given as arguments to `OPTIM` will apply only to this optimization step, whereas commands specified outside will act as default definitions for all subsequent `OPTIM` specifications.

The `OPTIM` keyword need not be specified if only one optimization step is required,

When only a machine-generated guess is available, `CASVB` will attempt to define a sequence of optimization steps that aims to maximize the likelihood of successful convergence (while minimizing CPU usage). To override this behaviour, simply specify one or more `OPTIM` keywords. The `ENDOPTIm` keyword marks the end of the specifications of an optimization step.

ALTErn

A loop over two or more optimization steps may be specified using:

`ALTERN`
Niter
...
`ENDALTERN`

The program will repeat the specified optimization steps until either all optimizations have converged, or the maximum iteration count, *Niter*, has been reached. The `ENDALTErn` keyword marks the end of the specification of an ALTERN loop.

Optional keywords for definitions of molecular symmetry and any constraints on the VB wavefunction

Keyword	*Meaning*

SYMELM	Various issues associated with symmetry-adapting valence bond wavefunctions are considered, for example, in: T. Thorsteinsson, D. L. Cooper, J. Gerratt and M. Raimondi, Theor. Chim. Acta **95**, 131 (1997).
	`SYMELM`
	label sign
	Initiates the definition of a symmetry operation referred to by *label* (any three characters). *sign* can be + or −; it specifies whether the total wavefunction is symmetric or antisymmetric under this operation, respectively. A value for *sign* is not always necessary but, if provided, constraints will be put on the structure coefficients to ensure that the wavefunction has the correct overall symmetry (note that the configuration list must be closed under the orbital permutation induced by *label* for this to be possible). The default for *label* is the identity.
	The operator is defined in terms of its action on the active MOs as specified by one or more of the keywords `IRREPS`, `COEFFS`, or `TRANS`. Any other keyword, including optional use of the `ENDSYMElm` keyword, will terminate the definition of this symmetry operator.
	`IRREPS`
	i_1 i_2 $\ldots$
	The list i_1 i_2 $\ldots$ specifies which irreducible representations (as defined in the CASSCF wavefunction) are antisymmetric with respect to the *label* operation. If an irreducible representation is not otherwise specified it is assumed to be symmetric under the symmetry operation.
	`COEFFS`
	i_1 i_2 $\ldots$
	The list i_1 i_2 $\ldots$ specifies which individual CASSCF MOs are antisymmetric with respect to the *label* operation. If an MO is not otherwise specified, it is assumed to be symmetric under the symmetry operation. This specification may be useful if, for example, the molecule possesses symmetry higher than that exploited in the CASSCF calculation.
	`TRANS`
	n_{dim} i_1 $\ldots i_{n_{dim}}$ c_{11} c_{12} $\ldots c_{n_{dim}n_{dim}}$
	Specifies a general $\mathrm{n}_{dim} \times n_{dim}$ transformation involving the MOs $i_1, \ldots i_{n_{dim}}$, specified by the c coefficients. This may be useful for systems with a two- or three-dimensional irreducible representation, or if localized orbitals define the CASSCF wavefunction. Note that the specified transformation must always be orthogonal.

ORBRel

In general, for a VB wavefunction to be symmetry-pure, the orbitals must form a representation (not necessarily irreducible) of the symmetry group. Relations between orbitals under the symmetry operations defined by `SYMELM` may be specified according to:

`ORBREL`
i_1 i_2 *label1 label2* ...

Orbital i_1 is related to orbital i_2 by the sequence of operations defined by the *label* specifications (defined previously using `SYMELM`). The operators operate right to left. Note that i_1 and i_2 may coincide. Only the minimum number of relations required to define all the orbitals should be provided; an error exit will occur if redundant `ORBREL` specifications are found.

(NO)SYMProj

As an alternative to incorporating constraints, one may also ensure correct symmetry of the wavefunction by use of a projection operator:

`(NO)SYMPROJ`
[$irrep_1$ $irrep_2$ ...]

The effect of this keyword is to set to zero the coefficients in unwanted irreducible representations. For this purpose, the symmetry group defined for the CASSCF wavefunction is used (always a subgroup of D_{2h}). The list of irreps in the command specifies which components of the wavefunction should be kept. If no irreducible representations are given, the current wavefunction symmetry is assumed. In a state-averaged calculation, all irreps are retained for which a non-zero weight has been specified in the wavefunction definition. The `SYMPROJ` keyword may also be used in combination with constraints.

FIXOrb

`FIXORB`
i_1 i_2 ...

This command freezes the orbitals specified in the list i_1 i_2 ... to that of the starting guess. Alternatively the special keywords `ALL` or `NONE` may be used. These orbitals are eliminated from the optimization procedure, but will still be normalized and symmetry-adapted according to any `ORBREL` keywords given.

FIXStruc

`FIXSTRUC`
i_1 i_2 ...

Freezes the coefficients for structures i_1, i_2,.... Alternatively the special keywords `ALL` or `NONE` may be used. The structures are eliminated from the optimization procedure, but may still be affected by normalization or any symmetry keywords present.

DELStruc

`DELSTRUC`
i_1 i_2,...

Deletes the specified structures from the wavefunction. The special keywords `ALL` or `NONE` may be used. This specification should be compatible with the other structure constraints present, as defined by `SYMELM` and `ORBREL`.

ORTHCON

`ORTHCON`
key-1 ...
key-2 ...
...

The `ORTHCON` keyword initiates the input of orthogonality constraints between pairs/groups of valence bond orbitals. The sub-keywords *key-i* can be any of `ORTH`, `PAIRS`, `GROUP`, `STRONG` or `FULL`. Orthogonality constraints should be used with discretion. Note that orthogonality constraints for an orbital generated from another by symmetry operations (using the `ORBREL` keyword) cannot in general be satisfied. The `ENDORTHcon` keyword can be used to terminate the input of orthogonality constraints.

`ORTH` i_1 i_2 ...

Specifies a list of orbitals to be orthogonalized. All overlaps between pairs of orbitals in the list are set to zero.

`PAIRS` i_1 i_2 ...

Specifies a simple list of orthogonalization pairs. Orbital i_1 is made orthogonal to i_2, i_3 to i_4, etc.

`GROUP` *label* i_1 i_2 ...

Defines an orbital group to be used with the `ORTH` or `PAIRS` keyword. The group is referred to by *label* which can be any three characters beginning with a letter a–z. Labels defining different groups can be used together or in combination with orbital numbers in `ORTH` or `PAIRS`. i_1 i_2 ... specifies the list of orbitals in the group. Thus the combination `GROUP` AAA 1 2 `GROUP` BBB 3 4 `ORTH` AAA BBB will orthogonalize the pairs of orbitals 1-3, 1-4, 2-3 and 2-4.

`STRONG`

This keyword is short-hand for strong orthogonality. The only allowed non-zero overlaps are between pairs of orbitals ($2n-1$, $2n$).

`FULL`

This keyword is short-hand for full orthogonality and is mainly useful for testing purposes.

Optional keywords for wavefunction analysis

Keyword *Meaning*

CIWEIGHTS

For further details regarding the calculation of weights in `CASVB`, see T. Thorsteinsson and D. L. Cooper, J. Math. Chem. **23**, 105-26 (1998).

`CIWEIGHTS`
key1 key2 ... $[N_{\mathrm{conf}}]$

Prints weights of the CASSCF wavefunction transformed to the basis of nonorthogonal VB structures. For the *key* options see `VBWEIGHTS` below. Note that the evaluation of inverse overlap weights involves

	an extensive computational overhead for large active spaces. Weights are given for the total CASSCF wavefunction, as well as the orthogonal complement to Ψ_{VB}. The default for the number of configurations requested, N_{conf}, is 10. If $N_{\mathrm{conf}}=-1$ all configurations are included.
REPOrt	`REPORT` [... `ENDREPORT`] Outputs orbital/structure coefficients and derived information. The `ENDREPOrt` keyword can be used to mark the end of the specification of a report step.
(NO)SCORr	`(NO)SCORR` With this option, expectation values of the spin operators $(\hat{s}_{\mu}+\hat{s}_{\nu})^2$ are evaluated for all pairs of μ and ν. Default is `NOSCORR`. The procedure is described by: G. Raos, J. Gerratt, D. L. Cooper and M. Raimondi, Chem. Phys. **186**, 233–250 (1994); ibid, 251–273 (1994); D. L. Cooper, R. Ponec, T. Thorsteinsson and G. Raos, Int. J. Quant. Chem. **57**, 501–518 (1996). This analysis is currently only implemented for spin-coupled wavefunctions.
VBWEights	For further details regarding the calculation of weights in `CASVB`, see T. Thorsteinsson and D. L. Cooper, J. Math. Chem. **23**, 105-26 (1998). `VBWEIGHTS` *key1 key2 ...* Calculates and outputs weights of the structures in the valence bond wavefunction Ψ_{VB}. *key* specifies the definition of nonorthogonal weights to be used, and can be one of: `CHIRGWIN` Evaluates Chirgwin-Coulson weights (see: B. H. Chirgwin and C. A. Coulson, Proc. Roy. Soc. Lond. **A201**, 196 (1950)). `LOWDIN` Performs a symmetric orthogonalization of the structures and outputs the subsequent weights. `INVERSE` Outputs "inverse overlap populations" as in G. A. Gallup and J. M. Norbeck, Chem. Phys. Lett. **21**, 495–500 (1973). `ALL` All of the above. `NONE` Suspends calculation of structure weights. The commands `LOWDIN` and `INVERSE` require the overlap matrix between valence bond structures, so that some additional computational overhead is involved.

Optional keywords for further general options

Keyword	*Meaning*

PREC

`PREC`
iprec iwidth

Adjusts the precision for printed quantities. In most cases, *iprec* simply refers to the number of significant digits after the decimal point. Default is *iprec*=+8. *iwidth* specifices the maximum width of printed output, used when determining the format for printing arrays.

PRINT

`PRINT`
i_1 i_2 ...

Each number specifies the level of output required at various stages of the execution, according to the following convention:

-1	No output except serious, or fatal, error messages.
0	Minimal output.
1	Standard level of output.
2	Extra output.

The areas for which output can be controlled are:

i_1	Print of input parameters, wavefunction definitions, etc.
i_2	Print of information associated with symmetry constraints.
i_3	General convergence progress.
i_4	Progress of the 2nd-order optimization procedure.
i_5	Print of converged solution and analysis.
i_6	Progress of variational optimization.
i_7	File usage.

For all, the default output level is +1. If $i_5 \geq 2$ VB orbitals will be printed in the AO basis (provided that the definition of MOs is available).

SHSTRUC

Prints overlap and Hamiltonian matrices between VB structures.

STATS

`STATS`

Prints timing and usage statistics.

Input example

```
&seward
symmetry
x␣y
basis␣set
c.sto-3g....
c␣0␣0␣-0.190085345
end␣of␣basis
basis␣set
h.sto-3g....
h␣0␣1.645045225␣1.132564974
```

```
end␣of␣basis
&scf
occupied
3␣0␣1␣0
&rasscf
inactive
1␣0␣0␣0
ras2
3␣1␣2␣0
nactel
6␣0␣0
lumorb
&casvb
```

Viewing and plotting VB orbitals

In many cases it can be helpful to view the shape of the converged valence bond orbitals, and Mocas therefore provides two facilities for doing this. For the Molden program, an interface file is generated at the end of each `CASVB` run (see also Section **??**). Alternatively a `CASVB` run may be followed by `RASSCF` to get orbitls (Section 8.30) and `GRID_IT` with the VB specification (Section 8.16), in order to generate a three-dimensional grid, for viewing, for example, with `MOLCAS GV` program.

8.4 CCSDT

`CCSDT` performs the iterative single determinant CCSD procedure for open shell systems and the noniterative triple contribution calculation to the CCSD energy. It is a shell-script/exec that is implemented in the *MOLCAS* framework as if it was a program module. The shell automates the calling sequence to programs required to determine coupled cluster type wave functions. The `CCSDT` program does the necessary reorganization, then performs the CCSD iterative procedure, and optionally calculates the triple excitation contribution. For further details the reader is referred to the sections 4.2.12 and 5.4 of the tutorials and examples manual.

8.4.1 Dependencies

`CCSDT` requires previous run of the `RASSCF` program to produce orbital energies, Fock matrix elements, wave function specification, and some other parameters stored in file *`JOBIPH`*. `RASSCF` program should be run with the "OUTORBITALS CANONICAL" to produce the *`JOBIPH`* file in proper format. `CCSDT` also requires transformed integrals produced by `MOTRA` and stored in the files *`TRAONE`* and *`TRAINT`*.

It is well known that the CCSD procedure brings the spin contamination into the final wave function $|\Psi\rangle$ even in the case where the reference function $|\Phi\rangle$ is the proper spin eigenfunction. The way how to reduce the spin contamination and mainly the number of independent amplitudes is to introduce the spin adaptation.

Besides the standard nonadapted (spinorbital) CCSD procedure this program allows to use different levels of spin adaptation of CCSD amplitudes (the recommended citations are Refs. [89, 90]):

- DDVV T2 adaptation.
  This is the most simple and most universal scheme, in which only the dominant part of T2 amplitudes, namely those where both electrons are excited from *doubly occupied (inactive)* to *virtual (secondary)* orbitals. The remaining types of amplitudes are left unadapted, i.e. in the spinorbital form. This alternative is an excellent approximation to the full adaptation and can be used for any multiplet.

- Full T1 and T2 adaptation (only for doublet states yet).
  In this case full spin adaptation of all types of amplitudes is performed. In the present implementation this version is limited to systems with the single unpaired electrons, i.e. to the doublet states only.

Besides these two possibilities there are also available some additional partial ones (see keyword ADAPTATION in Section 8.4.3). These adaptations are suitable only for some specific purposes. More details on spin adaptation in the CCSD step can be found in Refs. [90, 89, 91]. The current implementation of the spin adaptation saves no computer time. A more efficient version is under development.

The noniterative triples calculation can follow these approaches:

- CCSD + T(CCSD) - according to Urban et. al. [92]
- CCSD(T) - according to Raghavachari el. al. [93]
- CCSD(T) - according e.g. to Watts et. al. [14]

Actual implementation and careful analysis and discussion of these methods is described in Ref. [15], which is a recommended reference for this program.

The first alternative represents the simplest noniterative T3 treatment and contains only pure $\langle T3|WT2\rangle$ term. Second possibility represents the well known extension to the first one by the $\langle T3|WT1\rangle$ term (W is the two electron perturbation). For closed shell systems this is the most popular and most frequently used noniterative triples method. For single determinant open shell systems, described by the ROHF reference function standard (Raghavachari et. al.) method needs to be extended by the additional fourth order energy term, namely $\langle T3|UT2\rangle$ (U is the off-diagonal part of the Fock operator).

In contrast to the iterative CCSD procedure, noniterative approaches are not invariant with respect to the partitioning of the Hamiltonian. Hence, we obtain different results using orbital energies, Fock matrix elements or some other quantities in the denominator. According to our experiences [15], diagonal Fock matrix elements in the denominator represent the best choice. Using of other alternatives requires some experience. Since the triple excitation contribution procedure works strictly within the restricted formalism, resulting noniterative triples contributions depend also on the choice of the reference function. However, differences between this approach (with the reference function produced by a single determinant RASSCF procedure and the diagonal Fock matrix elements considered in the denominator) and the corresponding invariant treatment (with the semicanonical orbitals) are found to be chemically negligible.

For noniterative T3 contribution both non-adapted (spin-orbital) and spin-adapted CCSD amplitudes can be used. For more details, see Ref. [15].

8.4.2 Files

Input files

File	*Contents*
JOBIPH	This file is the `RASSCF` interface. It contains wavefunction specification, orbital energies and some other important parameters. For more information, see 8.30.2.
TRAONE	This file contains the transformed one-electron integrals produced by program `MOTRA`.
*TRAINT**	This file contains the transformed two-electron integrals produced by program `MOTRA`.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
TEMP000	Help file for creating names.
TEMPxxx	Sequential files - one for each *xxx* index in given symmetry. They contain integrals together with the remaining 3 indices. TEMP's are created for each symmetry combination, so they are not of the fixed size during the CCSORT run.
TEMPDA1	Direct access file containing integrals $\langle < ab \vert pq\rangle$ for given symmetry of the index *a*. This file is not produced in calculations employing C_1 point group symmetry.
TEMPDA2	A direct access file containing integrals $\langle am \vert pq\rangle$ for given symmetry index *a*. It is produced for any point group.
INPDAT	This file contains wavefunction specification, orbital energies and some other informations extracted mainly from *JOBIPH* file.
INTSTA	This file contains smaller integral arrays (up to O^2V^2 size) $f_{pq}\alpha\alpha, \beta\beta$ $\langle ij \vert\vert kl\rangle \alpha\alpha\alpha\alpha, \beta\beta\beta\beta, \alpha\beta\alpha\beta$ $\langle ka \vert\vert ij\rangle \alpha\alpha\alpha\alpha, \beta\beta\beta\beta, \alpha\beta\alpha\beta, \beta\alpha\alpha\beta$ $\langle ab \vert\vert ij\rangle \alpha\alpha\alpha\alpha, \beta\beta\beta\beta, \alpha\beta\alpha\beta$

INTAB	This file contains the integrals $\langle ab\|\|pq\rangle$ for all $a \geq b$
INTA1-4	This files contains the integrals $\langle ab\|\|pq\rangle$ for all $a \geq b$ $\langle ai\|\|ef\rangle, \langle ai\|\|ej\rangle$ for all a - α, β
Temp17-37	number of these files depends on the selected extrapolation given in the input file. There are 17 ... 28+2*diis+1 files. The size of Temp's is changing during the CCSD run, but usually they contain from $\frac{1}{4}$ to $\frac{5}{4}O^2V^2$ integrals.

Output files

File	*Contents*
RSTART	file with CC amplitudes and CC energy. The name of the file can be changed using keyword RESTART. It contains restart information, like T1aa,T1bb,T2aaaa,T2bbbb,T2abab, CC energy and the number of iterations.
T3hfxyy	These files contain integrals of $\langle ia\|bc\rangle$ type where x represents the symmetry and yy the value of the given index i. The number of these files is equal to the number of α occupied orbitals (*inactive + active.*)

8.4.3 Input

The input for each module is preceded by its name like:

```
␣&CCSDT
```

Keyword	*Meaning*
TITLE	This keyword starts the reading of title lines, with the number of title lines limited to 10. Reading the input as title lines is stopped as soon an the input parser detects one of the other keywords, however only ten lines will be accepted. This keyword is *optional.*
CCSD	This keyword specifies that only CCSD calculation will follow and the integrals will be prepared for the CCSD procedure only. This keyword is *optional.* (Default=OFF)

CCT	This keyword specifies that after CCSD calculation also noniterative T3 step will follow. For such calculations this key must be switched on. The integrals for the triple contribution calculation will then be prepared. This keyword is *optional.* (Default=ON)
ADAPtation	The parameter on the following line defines the type of spin adaptations of CCSD amplitudes. 0 - no spin adaptation - full spinorbital formalism 1 - T2 DDVV spin adaptation 2 - T2 DDVV + T1 DV spin adaptation (only recommended for specific purposes, since the adaptation of T1 included incompletely) 3 - full T2 and T1 spin adaptation (in current implementations limited to doublets only) 4 - full T2 adaptation without SDVS coupling (for doublets only) This keyword is *optional.* (Default=0)
DENOminators	The parameter on the following line specifies the type of denominators that will be used in the CCSD procedure. 0 - diagonal Fock matrix elements (different for α and β spins) 1 - spin averaged diagonal Fock matrix elements - $\frac{f_{\alpha\alpha}+f_{\beta\beta}}{2}$ 2 - orbital energies In some cases alternatives 1 and 2 are identical. For nonadapted CCSD calculations the resulting CCSD energy is invariant with respect to the selection of denominators. However, convergence may be affected. In the present implementation a symmetric denominators (i.e. the input 1 or 2) should be used for spin adapted CCSD calculations. This keyword is *optional.* (Default=0)
SHIFts	Following line contains *socc* and *svirt* levelshift values for occupied and virtual orbitals respectively. Typical values are in the range 0.0 - 0.5 (in *a.u.*) $dp(occ) = dp(occ) - socc$ $dp(virt) = dp(virt) + svirt$ For spin adaptations 3 and 4 only inactive (D) and active (V) orbitals will be shifted, due to the character of the adaptation scheme. For other cases all orbitals are shifted. This keyword is *optional.* (Defaults: $socc = 0.0, svirt = 0.0$)
TRIPles	The parameter on the following line specifies the type of noniterative triples procedure. There are three different types of perturbative triples available (see Section 8.4). 0 - CCSD approach (no triples step) 1 - CCSD+T(CCSD) according to Urban et. al [92] 2 - CCSD(T) according to Raghavachari et. al. [93] 3 - CCSD(T) according e.g. to Watts et. al. [14] This keyword is *optional.* (Default=3)
T3DEnominators	The parameter on the following line specifies the type of denominators that will be used in noniterative triples procedure. 0 - diagonal Fock matrix elements (different for α and β spins)

	1 - spin averaged diagonal Fock matrix elements - $\frac{f_{\alpha\alpha}+f_{\beta\beta}}{2}$ 2 - orbital energies In some cases alternatives 1 and 2 are identical. This keyword is *optional.* (Default=0)
T3SHifts	The following line contains *socc* and *svirt* levelshift values for occupied and virtual orbitals respectively. Typical values are in the range 0.0 - 0.5 (in *a.u.*) $dp(occ) = dp(occ) - socc$ $dp(virt) = dp(virt) + svirt$ In contrast to the iterative CCSD procedure, in noniterative T3 step results are not invariant with respect to the denominator shifting. It is extremely dangerous to use any other than 0.0 0.0 shifts here, since resulting T3 energy may have no physical meaning. This keyword may be useful only in estimating some trends in resulting energy, however, using of default values is strongly recommended. This keyword is *optional.* (Defaults: $socc = 0.0, svirt = 0.0$)
ITERations	This keyword is followed on the next line by the maximum number of iterations in the CCSD procedure. In the case of the RESTART run this is the number of last allowed iteration, since counting of iterations in RESTART run starts from the value taken from the ***RSTART*** file. This keyword is *optional.* (Default=30)
ACCUracy	The real value on the following line defines the convergence criterion on CCSD energy. This keyword is *optional.* (Default=1.0d-7)
END of input	This keyword indicates that there is no more input to be read. This keyword is *compulsory.*
EXTRapolation	This keyword switches on the DIIS extrapolation. This keyword is followed by two additional parameters on the next line *n1* and *n2.* *n1* - specifies the first iteration, in which DIIS extrapolation procedure will start for the first time. This value must not be less then *n2*, recommended value is 5-7. *n2* - specifies the size of the DIIS procedure, i.e. the number of previous CCSD steps which will be used for new prediction. In the present implementation *n2* is limited to 2-4. This keyword is *optional.* (Default=OFF)
PRINt	The parameter on the next line specifies the level of output printing 0 - minimal level of printing 1 - medium level of printing 2 - full output printing (useful for debugging purposes) This keyword is *optional.* (Default=0)
LOAD	This keyword is followed by the line which specifies the name of the CCSD amplitudes and energy file. The default name is ***RSTART***, but it can be changed in CCSD step using RESTART keyword. This keyword is *optional.* (Default=***RSTART***)

RESTart	This keyword defines the restart conditions and modifies the name of the file, in which restart information (CC amplitudes, CC energy and the number of iterations) is saved. On the following two lines there are control key *nn* and the name of restart information storing file *name*. *nn* - restart status key 0 - restart informations will be not saved 1 - restart informations will be saved after each iteration in *name*. 2 - restart run. CC amplitudes and energy will be taken from *name* file and the CCSD procedure will continue with these values as an estimate. *name* - specifies the restart information storing key. The name is limited to 6 characters. This keyword is *optional*. (Defaults: *nn*=1, *name*=RSTART)
IOKEy	This keyword specifies the input-output file handling. 1 - Internal Fortran file handling 2 - MOLCAS DA file handling The default (1) is recommended in majority of cases, since when calculating relatively large systems with low symmetry, the size of some intermediate files produced may become large, what could cause some troubles on 32-bit machines (2 GB file size limit).
MACHinetyp	This keyword specifies which type of matrix multiplication is preferred on a given machine. The following line contains two parameters *nn,limit*. *nn* =1 standard multiplication $A \times B$ is preferred *nn* =2 transposed multiplication $A^T \times B$ is preferred Parameter *limit* specifies the limit for using $A^T \times B$ multiplication, when *nn*=2. (It has no meaning for *nn*=1.) If *size(A)/size(B)* $\geq$ *limit* - standard multiplication is performed, *size(A)/size(B)* $<$ *limit* - transposed multiplication is performed. (*size(A,B)* - number of elements in matrix A,B). Recommended value for *limit* is 2-3. Using of transposed matrix (*nn*=2) multiplication may bring some computer time reduction only in special cases, however, it requires some additional work space. Default is optimal for absolute majority of cases. This keyword is *optional*. (Default=1).

Note, that CCSD and CCT keywords are mutually exclusive.

8.4.4 How to run closed shell calculations using ROHF CC codes

First of all it should be noted here, that it is not advantageous to run closed shell calculations using ROHF CC codes, since in the present implementation it will require the same number of arithmetical operations and the core and disk space like corresponding open shell calculations.

Since ROHF CC codes are connected to the output of `RASSCF` code (through the *`JOBIPH`* file), it is necessary to run closed shell Hartree-Fock using the `RASSCF` program. This can be done by setting the number of active orbitals and electrons to zero (also by including only doubly occupied orbitals into the active space; this has no advantage but increases the computational effort). to guarantee the single reference character of the wave function.

The CC program will recognize the closed shell case automatically and will reorganize all integrals in a required form. For more information the reader is referred to the tutorials and examples manual.

Below is an input file for HF$^+$ CCSD(T) calculation.

```
␣&CCSDT
Title
␣HF(+)␣CCSD(T)␣input␣example
CCT
Triples
3
```

8.5 CHCC

CHCC The CHCC is a Closed-Shell Coupled-Clusters Singles and Doubles program based exclusively on the Cholesky (or RI) decomposed 2-electron integrals aimed towards calculation of large systems on highly parallel architectures. Use of point-group symmetry is not implemented. Main advantage compared to the CCSDT module is *MOLCAS* is in its more efficient parallelization and dramatically lowered memory (and eventually disk) requirements.

8.5.1 Dependencies

CHCC requires previous run of the RHF SCF program to produce molecular orbitals and orbital energies stored in *RUNFILE*. The SCF program (as well as SEWARD) must be run in Cholesky/RI mode.

The algorithm used for almost complete elimination of the CHCC limits in calculated system size due to the computer memory bottleneck relies on blocking of the virtual orbitals. Number of blocks (further also referred to as the "large" segmentation, LARGE), N', should be as small as possible, because increasing of the segmentation brings in more CPU and I/O overhead. Futhermore, blocking can be "fine tuned" by, so called, "small" segmentation (SMALL), N", which affects only the (typically) most demanding $\approx O^2V^4$ scaling terms. The "large" segmentation can range from 1 to 32, "small" segmentation from 1 to 8, but their product, i.e. "large x small" must be lower then 64.

Selected blocking also determines the number of "independent" parallel tasks that must be executed in each iteration of the CCSD equations. In other words, particular segmentation predetermines the optimal number of computational nodes (i.e., if the best possible parallelization is desired). If the requested "large" segmentation is N', than N'^2 terms scaling as $\approx O^3V^3$ and 1/2 N'^2 terms scaling as $\approx O^2V^4$ result. Depending on which of these terms dominated in the calculations (O^3V^3 is more demanding for systems with large number of occupied orbitals and rather small basis set, while $\approx O^2V^4$ dominated for relatively large basis sets, i.e. large number of virtual orbitals), number of these task should be divisible by the number of computational nodes for optimal performance. To make it simple, as a rule of thumb, $N'^2/2$ should be divisible by the number of nodes, since the O^3V^3 are typically twice less expensive then the O^2V^4 step. Otherwise, any reasonable (i.e. the number of tasks is larger than the number of computational nodes, obviously) combination is allowed.

8.5.2 Files

Input files

File	*Contents*
`RUNFILE`	File for communication of auxiliary information.
`CHVEC*`	Files for storing Cholesky vectors.
`CHRED*`	File for storing index arrays for addressing Cholesky vectors.
`CHORST*`	File containing information about Cholesky decomposition procedure
`CHOR2F`	File containing the mapping between Cholesky vector storage and the canonical orbital ordering.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
`I0intg, I1inxx, I2xxxx, I3xxxx`	(ij|kl), (a'i|jk), (a'i|b'j) and (a'b'|ij) integrals with full occupied (i,j,k, ...) and segmented virtual indexes (a',b', ...).
`W3xxxxxx, W4xxxxxxxx`	(v'v'|v'o) and (v'v'|v'v') integrals where "v'" stands for segmented virtual and "o" for occupied index. Only generated when the keyword PRECALCULATE is active, otherwise, these integrals are generated and contracted "on-the-fly".

Output files

File	*Contents*
`L0xxxx, L1xxxx, L2xxxx`	MO-transformed Cholesky vectors
`T2xxxx`	T2 (ij,a'b') excitation amplitudes
`RstFil`	Communication file containing T1 amplitudes, restart informations, etc.

8.5.3 Input

The input for each module is preceded by its name like:

```
 &CHCC
```

Keyword	*Meaning*
TITLe	This keyword starts the reading of title lines, with the number of title lines limited to 10. Reading the input as title lines is stopped as soon as the input parser detects one of the other keywords, however only ten lines will be accepted. This keyword is *optional.*
FROZen	Integer on the following line specifies number of inactive occupied orbitals in the CCSD calculation. This keyword is *optional.* (Default=0)
DELEted	Integer on the following line specifies number of inactive virtual orbitals in the CCSD calculation. This keyword is *optional.* (Default=0)
LARGe	Integer on the following line specifies the main segmentation of the virtual orbitals. Value must be between 1 (no segmentation) and 32. Product of Large and Small segmentation must be lower than 64. This keyword is *optional.* (Default=1)
SMALl	Integer on the following line specifies the auxiliary segmentation of the virtual orbitals. Value must be between 1 (no segmentation) and 8. Product of Large and Small segmentation must be lower than 64. Small segmentation doesn't generate extra parallel tasks. This keyword is *optional.* (Default=1)
CHSEgmentation	Integer on the following line specifies the block size of the auxiliary (Cholesky/RI) index. Value must lower than the minimal dimension of the auxiliary index on each computational node. This keyword is *optional.* (Default=100)
MHKEy	Integer on the following line specifies if library BLAS (MHKEy=1) or hard-coded fortran vector-vector, matrix-vector and matrix-matrix manipulation is used. This keyword is *optional.* (Default=1)
NOGEnerate	This keyword specifies that the pre-CCSD steps (regeneration of integrals from the Cholesky/RI vectors, etc.) are skipped. This keyword is *optional.* (Default=OFF)
ONTHefly	This keyword specifies that all integral types scaling steeper then O^2V^2 are generated "on-the-fly" from the Cholesky/RI vectors. Use of this keyword leads to dramatically savings of the disk resources, but leads to significant arithmetic overhead. Keywords "ONTHefly" and "PRECalculate" are mutually exclusive. This keyword is *optional.* (Default=OFF)

PRECalculate	This keyword specifies that all integral are precalculated before the CCSD iterative procedure starts. Use of this keyword leads to significant consumption of the disk space, especially is single-processor runs. This keyword is *optional.* (Default=ON)
NODIstribute	This keyword (in combination with the "PRECalculate" keyword) specifies that all integral are stored on each computational node. In case of all integrals being stored on each node, extra permutation symmetry can be applied, thus leading to significant savings of the disk space. However, in case of massively parallel runs (i.e. more than ≈8 nodes), savings from keeping only subset of integrals required on particular node are more significant than savings due to permutational symmetry. This keyword is *optional.* (Default=OFF)
JOINlkey	The parameter on the following line specifies, which algorithm is used for precalculation and of the integrals in parallel run. In parallel runs, `SEWARD` produces AO Cholesky/RI vectors segmented in auxiliary index over parallel nodes. Depending on the network bandwidth and computational power of each node, different algorithms can lead to optimal performance. Following options are available: 0 - None: no cumulation of Cholesky/RI vectors is needed (debug only). 1 - Minimal: Cholesky/RI vectors are cumulated prior to integral precalculation. Low network bandwidth is required. 2 - Medium: O^2V^2 integrals are generated from local Cholesky/RI vectors and cumulated along with the Cholesky/RI vectors afterwards. Other integrals are calculated from cumulated intermediates. 3 - Full: All integrals are generated from local Cholesky/RI vectors and cumulated afterwards. High network bandwidth is required. This keyword is *optional.* (Default=2)
MAXIterations	Integer on the following line specifies maximum number of CCSD iteration This keyword is *optional.* (Default=40)
RESTart	This keyword specifies that CCSD calculation is restarted from previous run. This keyword is *optional* and is currently under development, thus disabled. (Default=OFF)
THREshold	Double precision floating point number on the following line specifies the convergence threshold for the CCSD correlation energy. This keyword is *optional.* (Default=1.0d-6)
PRINtkey	The integer on the following line specifies the print level in output 1 - Minimal 2 - Minimal + timings of each step of the CCSD iterations 10 - Debug This keyword is *optional.* (Default=1)
END of input	This keyword indicates that there is no more input to be read. This keyword is *compulsory.*

```
␣&CHCC␣&END
Title
Benzene␣dimer
Frozen
12
Deleted
0
Large
4
Small
2
CHSEgment
100
Precalculate
Join
2
Maxiter
50
Threshold
1.0d-6
Print
2
End␣of␣Input
```

8.6 `CHT3`

`CHT3` is a Closed-Shell Coupled-Clusters perturbative triples program based exclusively on the Cholesky (or RI) decomposed 2-electron integrals aimed towards calculation of large systems on highly parallel architectures. Use of point-group symmetry is not implemented. Main advantage compared to the `CCSDT` module is *MOLCAS* is in its more efficient parallelization and dramatically lowered memory (and eventually disk) requirements.

8.6.1 Dependencies

`CHT3` requires previous run of the `CHCC` Cholesky/RI based CCSD program to produce T1 and T2 excitation amplitudes stored in *`T2xxxx`* and *`RstFil`* files. The `CHCC` program (as well as `SEWARD` and `SCF`) must be run in Cholesky/RI mode.

The algorithm used for almost complete elimination of the `CHT3` limits in calculated system size due to the computer memory bottleneck relies on blocking of the virtual orbitals. Size of blocks is, unlike in `CHCC` program, determined automatically for optimal performance.

8.6.2 Files

Input files

File	*Contents*
`RUNFILE`	File for communication of auxiliary information.
`L0xxxx, L1xxxx, L2xxxx`	MO-transformed Cholesky vectors

T2xxxx	T2 (ij,a'b') excitation amplitudes
RstFil	Communication file containing T1 amplitudes, restart informations, etc.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
KMATAA, KMATBA, LMATAA, LMATBA	Temporary integral files

Output files

File	*Contents*

None

8.6.3 Input

The input for each module is preceded by its name like:

```
␣&CHT3
```

Keyword	*Meaning*
TITLE	This keyword starts the reading of title lines, with the number of title lines limited to 10. Reading the input as title lines is stopped as soon as the input parser detects one of the other keywords, however only ten lines will be accepted. This keyword is *optional.*
FROZEN	Integer on the following line specifies number of inactive occupied orbitals in the (T) calculation. This keyword is *optional.* (Default=0)
DELETED	Integer on the following line specifies number of inactive virtual orbitals in the (T) calculation. This keyword is *optional.* (Default=0)
LARGE	Integer on the following line specifies the main segmentation of the virtual orbitals used in previous CCSD run. This keyword is *compulsory.* (Default=None)

MHKEy	Integer on the following line specifies if library BLAS (MHKEy=1) or hard-coded fortran vector-vector, matrix-vector and matrix-matrix manipulation is used. This keyword is *optional.* (Default=1)
NOGEnerate	This keyword specifies that the pre-(T) steps (generation of integrals from the Cholesky/RI vectors, etc.) are skipped. This keyword can be used for restarting the (T) calculation if the required integrals were already generated. This keyword is *optional.* (Default=OFF)
NOTRiples	This keyword specifies that the post integral preparation steps, i.e. the real calculation of (T) contribution will not be done. Job can be restarted from this point using the NOGEnerate keyword. This keyword is *optional.* (Default=OFF)
ALOOp	Two integers on the following line specify first and last triplet of virtual orbitals blocks to be calculated in the first ("A loop") of the two parts of the (T) calculation. Using this keyword enables user to split the (T) calculation into separate jobs. Information about the total number of triplets in the "A loop" can be found in the output of the "preparation" step of the (T) program. Values -1, -1 mean, that the whole "A loop" is either executed or skipped, depending on the parameters of the BLOOp keyword. This keyword is *optional.* (Default=-1,-1)
BLOOp	Two integers on the following line specify first and last triplet of virtual orbital block to be calculated in the second ("B loop") of two parts of the (T) calculation. Using this keyword enables user to split the (T) calculation into separate jobs. Information about the total number of triplets in the "B loop" can be found in the output of the "preparation" step of the (T) program. Values -1, -1 mean, that the whole "B loop" is either executed or skipped, depending on the values of the ALOOp keyword. This keyword is *optional.* (Default=-1,-1)
PRINtkey	The integer on the following line specifies the print level in output 1 - Minimal 2 - Minimal + timings of each (T) step 10 - Debug This keyword is *optional.* (Default=1)
END of input	This keyword indicates that there is no more input to be read. This keyword is *compulsory.*

```
␣&CHT3
Title
Benzene␣dimer
Frozen
12
Large
4
ALOOp
20,120
BLoop
```

```
1,250
Print
2
End␣of␣input
```

8.7 CIISCMNG

This program is a driver to compute the necessary gradients, in order to optimize conical intersections or intersystem crossings. It is not directly called by the user, but instead it is invoked by SLAPAF whenever the appropriate keywords are used.

8.7.1 Description

When the CIOPtimization (see section 8.34.4) for conical intersections (CI) or ISCOptimization (see section 8.34.4) for intersystem crossing (ISC) keywords are used in a `SLAPAF` input, the module `CIISCMNG` is called. `CIISCMNG` will take care of producing the necessary gradients by running the appropriate modules, and then it will restart `SLAPAF`. Four possibilities can arise: CI or ISC search, by analytical or numerical means for producing the gradients. Even if the user has not to care about the way to call `CIISCMNG`, special care has to be taken when writing the inputs, depending on which kind of search is requested. For detailed examples see section 8.7.3. The `CIISCMNG` is designed to work with `RASSCF` or `CASPT2` only. See the example section (8.7.3) for more details.

8.7.2 Dependencies

`CIISCMNG` must be preceded by one (CI) or two (ISC) `RASSCF` or `CASPT2` runs.

8.7.3 Files

Input files

Apart from the standard input file `CIISCMNG` will use the following input file.

File	*Contents*
`RUNFILE`	File for communication of auxiliary information.

Output files

In addition to the standard output file `SLAPAF` will produce the following output files.

File	*Contents*

RUNFILE	File for communication of auxiliary information.
RUNFILE2	File for communication of auxiliary information of the ”ground state” in case of minimum energy cross point optimizations.

Input Examples

The four possible cases will be treated separately. For each one, an example of an input will be given, being hopefully self-explanatory. Please, take note that such inputs are given mostly to point out the compulsory parts. When a `CASPT2` reference energy is employed, the numerical method is used by default.

CI optimization with analytical gradients (8.7.4).

CI optimization with numerical gradients (8.7.5).

ISC optimization with analytical gradients (8.7.6).

ISC optimization with numerical gradients (8.7.7).

8.7.4 CI search, analytical gradients

Method not implemented, yet

8.7.5 CI search, numerical gradients

Method not implemented, yet

8.7.6 ISC search, analytical gradients

```
!export␣OldProject=$Project␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
>>>>␣SET␣MAXITER␣200␣<<<<<<
>>>>>>>>>>>>>>>>>>>␣DO␣while␣<<<<<<<<<<<<<<<<<<<<
␣&Seward
...
End␣of␣input
>>>>>>>>>>>␣if␣(␣iter␣=␣1␣)␣<<<<<<<<<<<<<<
␣&SCF
...
End␣of␣input
>>>>>>>>>␣␣endif␣<<<<<<<<<<<<<<<<<<<<<<<<
****************************************
!ln␣-fs␣$OldProject.OrdInt␣State1.OrdInt␣␣#COMPULSORY
!ln␣-fs␣$OldProject.OrdInt␣State2.OrdInt␣␣#COMPULSORY
!ln␣-fs␣$OldProject.OneInt␣State1.OneInt␣␣#COMPULSORY
!ln␣-fs␣$OldProject.OneInt␣State2.OneInt␣␣#COMPULSORY
!cp␣$OldProject.RunFile␣State1.RunFile␣␣␣␣#COMPULSORY
!cp␣$OldProject.RunFile␣State2.RunFile␣␣␣␣#COMPULSORY
!ln␣-fs␣State1.RunFile␣RUNST1␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
!ln␣-fs␣State2.RunFile␣RUNST2␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
****************************************
>>>␣export␣Project=State1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
>>>>>>>>>>>␣if␣(␣iter␣=␣1␣)␣<<<<<<<<<<<<<<
```

```
␣&RASSCF
Lumorb
Title
␣RASSCF␣INPUT␣First␣State
...
CIRoot
3␣3
1␣2␣3
1␣1␣1
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
>>>>>>>>>␣␣endif␣<<<<<<<<<<<<<<<<<<<<<<<<
>>>export␣␣Project=State1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
␣&RASSCF
JOBIPH
CIRESTART
Title
␣RASSCF␣INPUT␣First␣State
...
CIRoot
3␣3
1␣2␣3
1␣1␣1
RLXRoot
3
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣export␣Project=State2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
>>>>>>>>>>>␣if␣(␣iter␣=␣1␣)␣<<<<<<<<<<<<<<
␣&RASSCF
Lumorb
Title
␣RASSCF␣INPUT␣Second␣State
...
CIRoot
2␣2
1␣2
1␣1
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
>>>>>>>>>␣␣endif␣<<<<<<<<<<<<<<<<<<<<<<<<
>>>export␣␣Project=State2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
␣&RASSCF
JOBIPH
CIRESTART
Title
␣RASSCF␣INPUT␣Second␣State
...
CIRoot
2␣2
1␣2
1␣1
RLXRoot
2
End␣of␣input
!cp␣$Project.JobIph␣$Project.JobOld
>>>␣export␣Project=$OldProject␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
!cp␣State1.RunFile␣$OldProject.RunFile␣␣␣␣␣#COMPULSORY
␣&Slapaf
ISCOptimization␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣#COMPULSORY
3␣2␣␣␣␣␣␣#COMPULSORY,␣MUST␣CORRESPOND␣TO␣THE␣RLXROOT␣OF␣THE␣FIRST␣AND␣SECOND␣RASSCF␣INPUTS
Iterations
20
End␣of␣input
```

8.7.7 ISC search, numerical gradients

Method not implemented, yet

8.8 CPF

The CPF program generates SDCI, CPF[94], MCPF[95] or ACPF[96], wavefunctions from **one** reference configuration.

The CPF program is a modification to a CPF program written by P. E. M. Siegbahn and M. Blomberg (Institute of Physics, Stockholm University, Sweden).

The program is based on the Direct CI method[97], with the coupling coefficients generated by the Graphical Unitary Group Approach[98]–[99] (See program description for GUGA). CPF generates natural orbitals that can be fed into the property program to evaluate certain one electron properties. Also, the natural orbitals can be used for Iterative Natural Orbital calculations.

Orbital subspaces

The orbital space is divided into the following subspaces: Frozen, Inactive, Active, Secondary, and Deleted orbitals. Within each symmetry type, they follow this order. Their meaning is the same as explained in the GUGA and MOTRA sections, except that, in this case, there is only a single reference configuration. Therefore, the active orbitals in this case are usually only open shells, if any. Since explicit handling of orbitals is taken care of at the integral transformation step, program MOTRA, orbital spaces are not specified in the input, except when orbitals are frozen or deleted by the CPF program, rather than by MOTRA (which should normally be avoided).

8.8.1 Dependencies

The CPF program needs the coupling coefficients generated by the program GUGA and the transformed one and two electron integrals from the program MOTRA .

8.8.2 Files

Input files

The CPF program need the coupling coefficients generated by GUGA and the transformed integrals from MOTRA. The following is a list of the input files CPF needs

File	*Contents*
CIGUGA	Coupling coefficients from GUGA.
*TRAINT**	Transformed two electron integrals from MOTRA.

TRAONE	Transformed one electron integrals from `MOTRA`.
ONEINT	One-electron integrals used for charges, properties etc.
CPFVECT	Used as input only in restarted calculations.

Output files

`CPF` generates an two output files:

File	*Contents*
CPFORB	The natural orbitals from the CPF functional.
CPFVECT	The CI expansion coefficients. These may be used for restarting an unconverged calculation.

8.8.3 Input

8.8.4 `CPF`

This section describes the input to the `CPF` program in the MOLCAS program system. The input for each module is preceded by its name like:

```
␣&CPF
```

The first four characters of the keywords are decoded while the rest are ignored.

Also, out of the choices SDCI, CPF, MCPF or ACPF, precisely one must be used. See below.

Optional keywords

Keyword	*Meaning*
TITLE	After this keyword follows an arbitrary number of title cards. At most ten lines will, however, be printed in the output.
SDCI	Specifies that a SDCI calculation is to be performed. No additional input is required. Only one of the choices SDCI, CPF, MCPF or ACPF should be chosen.
CPF	Specifies that a CPF calculation is to be performed. No additional input is required. Only one of the choices SDCI, CPF, MCPF or ACPF should be chosen.

MCPF	Specifies that a Modified CPF calculation is to be performed. No additional input is required. This option is in fact the default choice, but it does no harm to choose it. Only one of the choices SDCI, CPF, MCPF or ACPF should be chosen.
ACPF	Specifies that an Average CPF calculation is to be performed. No additional input is required. Only one of the choices SDCI, CPF, MCPF or ACPF should be chosen.
RESTart	Restart the calculation from a previous calculation. No additional input is required.
THRPr	Threshold for printout of the wavefunction. All configurations with a coefficient greater than this threshold are printed in the final printout. The default is 0.05. The value is read from the line following the keyword.
ECONvergence	Energy convergence threshold. The update procedure is repeated until the energy difference between the last two iterations is less than this threshold. The default is 1.0e-8. The value is read from the line following the keyword.
PRINt	Print level of the program. Default is 5. The value is read from the line following the keyword.
MAXIterations	Maximum number of iterations in the update procedure. Default 20. The value is read from the line following the keyword. The maximum value of this parameter is 75.
FROZen	Specify the number of orbitals to be frozen in **addition** to the orbitals frozen in the integral transformation. The values are read from the line following the keyword. Default is 0 in all symmetries.
DELEted	Specify the number of orbitals to be deleted in **addition** to the orbitals deleted in the integral transformation. The valued are read from the line following the keyword. Default is 0 in all symmetries.
LOW	Specifies that this is a low spin case, i.e. the spin is less than the maximum possible with the number of open shells in the calculation. See Refs. [94, 95]. This requires special considerations.
MAXPulay	Maximum number of iterations in the initial stage. After that, DIIS extrapolation will be used. Default is 6.
LEVShift	Levelshift in the update procedure. Default is 0.3.

Input example

```
␣&CPF
Title
␣Water␣molecule.
␣1S␣frozen␣in␣transformation.
MCPF
```

8.9 General input structure. EMIL commands

This is a general guide to the input structure of the programs in the *MOLCAS* program system. All programs conform to the same conventions except where explicitly stated otherwise.

The programs are driven by keywords, which are either used without further information, or followed by additional specifications on the line(s) following the keyword, and is normally numeric in nature. *All numerical inputs are read in free format, note that in general MOLCAS will not be able to process lines longer than 120 characters.* The keywords can be given in mixed case (both upper and lower case are allowed). In the input stream you can insert comment lines anywhere, except between a keyword and the following additional specifications, with a comment line identified by an asterisk (*) in the first position on the line.

Most codes look at the first 4 characters of the keyword and ignores the rest. The entries in the lists of keywords below follow the standard that the significant characters are in upper case and larger than the nonsignificant characters. This do not imply that the keywords have to be typed in upper case; they can be typed freely in mixed case.

All inputs begin with a name of the program preceeding the keywords:

```
␣&PROGRAM
*␣here␣follows␣the␣keywords
```

where PROGRAM is the name of the *MOLCAS* module. The input listing is finished when a new program name, preceeded by the symbol &, is found (or the end of file or an EMIL command).

The following is an example of a list of keywords common to most of the programs:

Keyword	*Meaning*
TITLe	This keyword starts the reading of title line. The following line is treated as title line.

The programs only decode the first four characters of a keyword (except otherwise specifically indicated). For clarity it is however recommended to write the full keyword name. The keywords can be typed freely in upper, lower or mixed case.

An example for an input file used to run the `SCF` program follows:

```
␣&SCF
Title
␣Water␣molecule.␣Experimental␣equilibrium␣geometry
*␣The␣symmetries␣are:␣a1,␣b2,␣b1␣and␣a2.
Occupied
3␣1␣1␣0
*␣The␣ivo␣keyword␣prepares␣virtual␣orbitals␣for␣MCSCF.
Ivo
```

MOLCAS contains an input-oriented shell-script named `AUTO` that is implemented into the *MOLCAS* framework as if it was a program module. The shell will run the *MOLCAS* programs sequentially in the order they appear in the general input file. The script allows to orginaize loops (for structure optimization), and execute modules or commands conditionally.

In the first step of execution `AUTO` analyses user's input, cleans it (removes tabulation signs, DOS-style end-of-lines, etc.), and makes some preprocessing of the input.

The internal language used by `AUTO` is EMIL (Extended Molcas Input Language). It includes three different types of input commands:

- Sections with *MOLCAS* input.
- EMIL commands (a line started with > character)
- UNIX commands

8.9.1 Molcas input

EMIL allows to write *MOLCAS* input in a more compact way: user can omit &*END*, as well as a compulsory (in previous versions of molcas) keyword END OF INPUT. As soon as a new module (or UNIX, or EMIL command) is requested in a user input, the input for the module is terminated.

Also, it is possible to separate lines by ; sign, or by = sign (to create a pair *keyword* = *value*). In some rare occasions signs ; and = are used in the input for a molcas module. In order to keep these symbols unchanged, user can mark a part of an input, containing these symbols, by EMIL commands >> *verbatim* and >> *endverbatim*.

It means that the input:

```
 &SCF &END
CHARGE
 1
End of input
 &ALASKA &END
End of input
 &SLAPAF &END
End of input
```

could be written as:

```
 &SCF; CHARGE=1
 &ALASKA;  &SLAPAF
```

User can comment parts of input, by using ∗ at the beginning of line, or use C-style comments (/ ∗ ... ∗ /) to comment several lines.

Including in the input file a UNIX command preceded by an exclamation mark allows to execute the command during the execution of `AUTO`. For instance the listing command !ls -ls. One can use one or two exclamation marks, the difference being that with two the command will only be executed on the master node, i.e. in serial executions there's no difference between the two forms. Note, that UNIX command export is not allowed in `AUTO`. User should use EMIL command EXPORT instead.

8.9.2 EMIL commands

EMIL commands can be written in a short form:

```
> KEY [VALUE]
```

or in a nice form:

```
>>>>>>>>>>  KEY  [VALUE]  <<<<<<<<<
```

Here is a list of EMIL commands:

Command	*Purpose*
>> EXPORT A=B	a command to set environment variable A to value B
>> EXIT	a command to terminate execution. An optional value for this command is the return code (default value is 0)
>> INCLUDE file	a command to include a file into the input A compulsory value for this command is the filename.
>> UNIX	a command to execute a UNIX command. This EMIL command is an alias for '!'.
>> LINK	a command to make a link between two files. The command is similar to '!ln -s FILE1 FILE2' but can be used also in parallel environment. >>LINK -FORCE is a portable implementation of '!ln -fs FILE1 FILE2'.
>> COPY	a command to make a copy. The command is similar to '!cp -f FILE1 FILE2' but can be used also in parallel environment. Note that EMIL command does not allow to use masks in the command.
>> RM	a command to delete a file. The command is similar to '!rm -f FILE' but can be used also in parallel environment. Note that EMIL command does not allow to use masks in the command.
>> EVAL A=B	evaluate a numerical value

Keywords to organize loops in input, and execute modules conditionally:

Command	*Purpose*
>> DO WHILE	a command to start a loop. The loop should be terminated by SLAPAF or LOOP module, followed by ENDDO command
>> FOREACH A in (B, C, D)	a command to loop when the value of A is in the coma separated list. The list also can be written in the format "From .. To". Note that variable in the loop must be uppercased.

>> ENDDO	a command to finish the loop. If last module (before ENDDO command) returns 1 - the loop will be executed again (if number of iterations is less than MAXITER). If the return code is equal to 0 the loop will be terminated.
>> IF (ITER = N)	- a command to make conditional execution of modules/commands on iteration N (N possibly could be a space separated list)
>> IF (ITER != N)	- a command to skip execution of modules/commands on iteration N
>> IF ($VAR = N)	- a command to make conditional execution if $VAR value equals to N (if statement terminated by ENDIF command)
>> IF ($VAR = N) GOTO JUMP	- a command to make conditional goto to a label JUMP
>> LABEL JUMP	- a command to define a label.Note! Only forward jumps are allowed.
>> ENDIF	terminate IF block. Note nested if's are not allowed.

`AUTO` automatically stops calculation if a module returns a returncode higher than 0 or 1. To force `AUTO` to continue calculation even if a returncode equal to 16 (which is a return code for non-convergent calculation) one should set environment variable MOLCAS_TRAP='OFF'.

`SLAPAF` returns a special return code in the case of converged (non converged) geometry. So, to organize a structure calculation one should place the call to `SLAPAF` as a last statement of loop block. The summary of geometry optimization convergence located in a file *`$Project.structure`*. The programs following a geometry optimization will automatically assume the optimized geometry and wave function. Any new `SEWARD` calculation after an optimization (minimum or transition state) will disregard the input coordinates and will take the geometry optimized by `AUTO`.

It is also possible to use a special dummy program `LOOP` to organize infinite loops, or loops terminated by the counter (set be MOLCAS_MAXITER)

SET - an auto command to change settings of the script

Command	*Purpose*
>> SET MAXITER N <<	set maximum number of loop iterations to N. This option is obsolete and can be substituted by $>> exportMOLCAS\_ITER =$
>> SET OUTPUT SCREEN <<	redirect output (in loops) to screen. This option is obsolete.
>> SET OUTPUT FILE <<	redirect output (in loops) to a set of files in WorkDir. This option is obsolete.
>> SET OUTPUT OVER <<	skip output during structure loops, and print only last iteration. This option is obsolete.

Verbatim input.

If an input for a module must contain special symbols, such as ; or =, user can mark a corresponding part of the input by EMIL command *VERBATIM*

Command	*Purpose*
>> VERBATIM <<	start verbatim input
>> END VERBATIM <<	finish verbatim input

Below are different input examples.

The first example shows the procedure to perform first a `CASSCF` geometry optimization of the water molecule, then a numerical hessian calculation on the optimized geometry, and later to make a `CASPT2` calculation on the optimized geometry and wave function. Observe that the position of the `SLAPAF` inputs controls the data required for the optimizations.

```
*
*␣␣␣␣Start␣Structure␣calculation
*
>>EXPORT␣MOLCAS_ITER=50
␣&GATEWAY
coord
Water.xyz
BASIS␣=␣ANO-S
>>>>>>>>>>>>>␣Do␣while␣<<<<<<<<<<<<
␣&SEWARD
>>>>>>>>␣IF␣(␣ITER␣=␣1␣)␣<<<<<<<<<<
␣&SCF
>>>>>>>␣␣ENDIF␣<<<<<<<<<<<<<<<<<<<<
␣&RASSCF
Title
␣H2O␣ANO(321/21).
Nactel␣␣␣=␣6␣␣0␣␣0
Spin␣␣␣␣␣=␣1
Inactive␣=␣1␣␣0␣␣0␣␣0
Ras2␣␣␣␣␣=␣3␣␣2␣␣1␣␣0
␣&ALASKA;␣&SLAPAF
>>>>>>>>>>>>>␣␣ENDDO␣␣<<<<<<<<<<<<<<
␣&CASPT2
Maxit␣=␣20
Lroot␣=␣1
␣&GRID_IT
```

Another example demonstarte a possibility to use loops. SCF module will be called twice - first time with BLYP functional, second time with B3LYP functional.

```
*------------------------------------------------------
␣&GATEWAY
coord
$MOLCAS/Test/input/C2H6.xyz
basis
ANO-S-VDZ
group
y␣xz
*------------------------------------------------------
␣&SEWARD
Title
Ethane␣DFT␣test␣job
```

```
*------------------------------------------------------
>>foreach␣DFT␣in␣(BLYP,␣B3LYP␣)
␣&SCF␣;␣KSDFT␣=␣$DFT
>>enddo
*------------------------------------------------------
The␣next␣examples␣calculates␣HF␣energy␣for␣the␣several␣structures:
*␣modify␣coordinates␣in␣place
>>foreach␣DIST␣in␣(1.0,␣2.0,␣20.0)
␣&GATEWAY
Coord
2
hydrogen␣molecule
H␣0␣0␣0
H␣$DIST␣0␣0
BASIS=␣ANO-S-MB
GROUP=␣C1
␣&SEWARD
␣&SCF
UHF
SCRAMBLE=0.3
>>enddo
*␣incremental␣change␣of␣coordinates
>>export␣DIST=1.0
>>foreach␣L␣in␣(␣1␣..␣3␣)
>>eval␣DIST=$DIST+0.1
␣&GATEWAY
Coord
2
hydrogen␣molecule
H␣0␣0␣0
H␣$DIST␣0␣0
BASIS=␣ANO-S-MB
GROUP=␣C1
␣&SEWARD
␣&SCF
>>enddo
*␣different␣coordinate␣files
>>foreach␣COO␣in␣(␣000,␣001,␣002)
␣&GATEWAY
Coord␣=␣H2$COO.xyz
BASIS=␣ANO-S-MB
GROUP=␣C1
␣&SEWARD
␣&SCF
>>enddo
```

8.9.3 Use of shell parameters in input

The MOLCAS package allows the user to specify parts or variables in the the input file with shell variables, which subsequently are dynamically defined during execution time. **Note:** the shell variable names must be in upper case. Find below a simple example where a part of the H$_2$ potential curve is computed. First, the script used to run the calculation:

```
#! /bin/sh
#
Home=‘pwd‘ ;                     export Home
Project=H2 ;                     export Project
WorkDir=/tmp/$Project ;          export WorkDir
#
# Create workdir and cd to it
#
rm -fr $WorkDir
mkdir $WorkDir
```

```
#
# Loop over distances
#
for R in 0.5 0.6 0.7 0.8 0.9 1.0
do
   export R
   molcas $Home/$Project.input > $Home/$Project-$R-log 2> $Home/$Project-$R-err
done
#
# Cleanup WorkDir
#
rm -fr $WorkDir
```

In this sh shell script we have arranged the call to the *MOLCAS* package inside a loop over the various values of the distances. This value is held by the variable $R which is exported every iterations. Below is the input file used, note that the third cartesian coordinate is the variable $R.

```
␣&SEWARD
Symmetry
␣x␣y␣z
Basis␣set
H.sto-3g....
H␣␣␣0.000␣␣␣0.000␣␣␣$R
End␣of␣basis
End␣of␣input
␣&SCF
```

8.10 *MOLCAS*-7 Flowchart

8.11 ESPF (+ QM/MM INTERFACE)

8.11.1 Description

The ElectroStatic Potential Fitted (ESPF) method adds some one-electron operators to the core Hamiltonian in order to compute the interaction between the charge distribution in *MOLCAS* and *any* external electrostatic potential, field, and field derivatives. Both energy and gradient computations are available: they need one call to `ESPF` right after running `SEWARD`. This plugin to *MOLCAS* is especially devoted for hybrid quantum mechanics/molecular mechanics (QM/MM) computations, in which an extended molecular system is divided into two subsystems: the 'reactive' center treated with QM and the surroundings qualitatively treated with an empirical atomic forcefield. The current implementation uses a modified version of the `TINKER` program as MM code. In order to obtain the modified `TINKER` code, see: `http://sites.univ-provence.fr/lcp-ct/ferre/nf_tinker_qmmm.html`.

The `ESPF` plugin can also be used to compute some atomic point charges fitted to the electrostatic potential.

NOTE that no symmetry is allowed since the external potential usually breaks the one given in `SEWARD`.

NOTE you must always run together `SEWARD` + `ESPF`. If not, very strange results may happen, due to interactions counted twice or more !

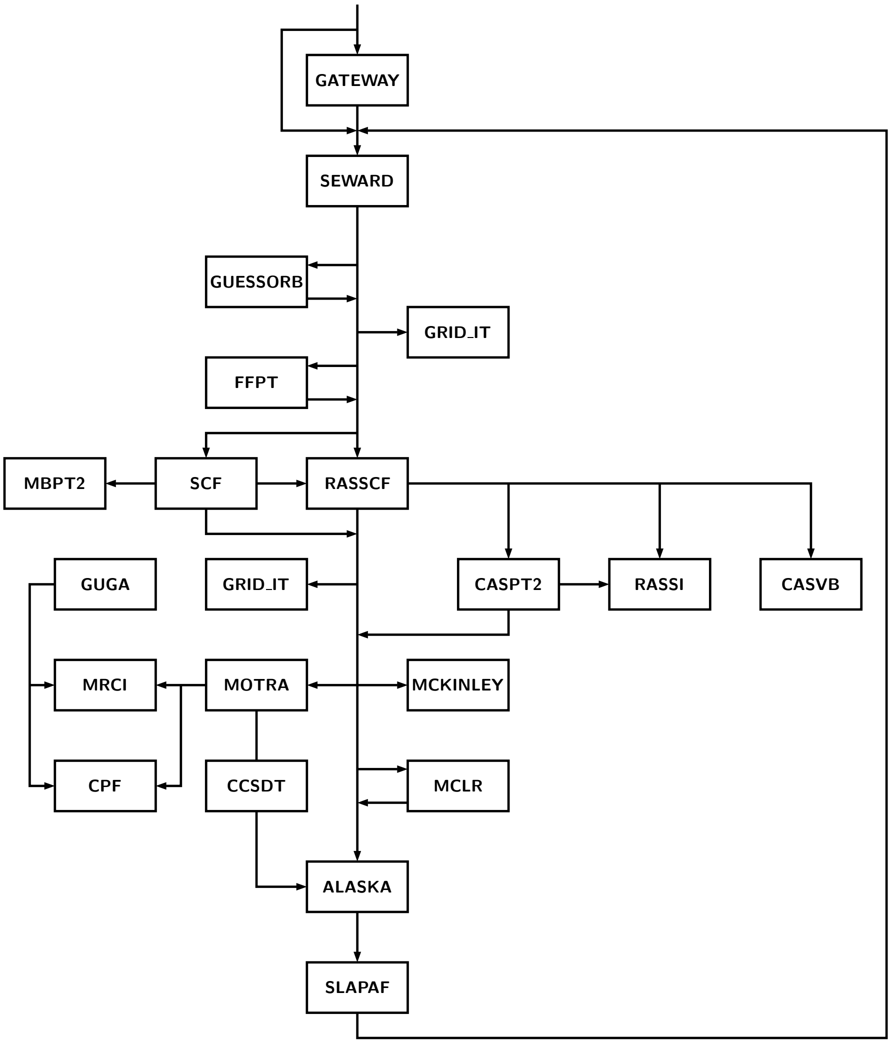


Figure 8.1: Program module dependencies flowchart for MOLCAS.

All details about the ESPF method can be found within the following reference: CPL 356 (2002) 331. However, few words can be said about the current QM/MM implementation. First both QM and MM atoms can be given in *MOLCAS*. MM atoms are specified in `SEWARD` using the following syntax:

```
Basis␣set
X.....␣/␣MM
␣␣name␣␣x␣␣y␣␣z␣␣Angstrom
Charge
␣␣q
End␣of␣basis
```

where q gives its MM point charge value, name is its name and x, y, z are its coordinates given in au or in Å if the ANGSTROM keyword is given. This way, a MM atom is simply a QM atom without basis function and with a non-integer atomic charge. Actually this possibility is very convenient when some link atoms (see below) positions must be constrained.

Now let's recall the *basics* of a QM/MM calculation. The Hamiltonian of the full QM/MM system is divided into three terms

$$H = H_{QM} + H_{MM} + H_{QM/MM} \tag{8.1}$$

the first one describes the QM part as it would be *in vacuo*, the second one describes the surroundings using a classical MM forcefield and the last one deals with the interactions between the QM and the MM subsystems. In its usual formulation, the last term is (for q point charges interacting with N nuclei and n electrons):

$$H_{QM/MM} = \sum_{a=1}^{q}\sum_{b=1}^{N}\frac{Q_a Z_b}{R_{ab}} - \sum_{a=1}^{q}\sum_{i=1}^{n}\frac{Q_a}{r_a} + \sum_{a=1}^{q}\sum_{b=1}^{N}E_{ab}^{vdw} + E^{bonded} \tag{8.2}$$

The first two terms deal with the electrostatic interactions between the quantum nuclei and electrons, and the classical point charges: the quantum wavefunction is polarized by these point charges. Actually the ESPF method replaces this direct interaction with an approximated one. A short-range van der Waals term is added (van der Waals parameters are assigned to all the atoms - both QM and MM). If the frontier between the two subsystems involves a bond, some empirical bonded terms may also be used. For the sake of simplicity, the standard MM parameters are kept unchanged for the MM atoms but should be modified (or calculated) for the QM atoms (e.g. it may be necessary to fit the QM van der Waals parameters).

The usual forcefields use the "1-4 condition" to separate the bonded interactions (stretching, bending, torsion) from the non-bonded ones (electrostatic and vdw). This means than the non-bonded potentials are applied only if atoms are separated by 3 bonds or more. For the QM/MM interactions, this procedure is kept with the exception that all the QM atoms experience the MM point charges without exceptions (the QM/MM frontier case is considered later). Thus, all the MM and QM/MM classical interactions are calculated using the MM forcefield, while the QM/MM electrostatic interactions are included in the QM part of the calculation.

When no bond are involved between the QM and the MM parts, the QM/MM frontier is obvious and only the electrostatic and vdw interactions are taken into account. However, if one or several chemical bonds exist, the definition of a smooth but realistic frontier is needed. Several schemes, more or less sophisticated, have been proposed. In the current implementation, only the most basic one, the **Link atom** (LA) approach is included. In the LA approach, every QM/MM bond that should be cut is saturated with a monovalent atom - most often a hydrogen atom - on the QM side. The position of a link atom is often restrained: frozen distance from the corresponding QM frontier atom and always on the segment defined by the two frontier atoms. From the macromolecular point of view, these link atoms do not exist → they should not interact with the MM part. However, this leads to severe overpolarization of the frontier, due to unbalanced interactions. It was found better to allow interactions between these link atoms and the MM point charges. To remove problems that may arise from too strong interactions between a link atom and the closest MM point charges, these point charges may be set to zero. In a protein, this procedure is mainly justified if the MM frontier atom is an α carbon (Amber or Charmm-typed forcefields usually set these point charges close to zero).

Usually, the LA position is constrained, e.g. to stay on the line connecting the QM and the MM atoms. The Morokuma's scheme is actually used for achieving such constraints.

Obviously, during a geometry optimization job, a *MOLCAS* step costs as hundreds of `TINKER` steps. Thus it is very convenient to use the microiterations technique, that is converging the

MM subsystem geometry every *MOLCAS* step. This is the default, but it can be modified directly within the `TINKER` keywords file. In order to improve the convergence, an improved QM/MM hessian can be built in `SLAPAF` using its rHidden keyword (note that adding the keyword Cartesian may help too).

8.11.2 Dependencies

The `ESPF` program depends on `SEWARD` for modifying the core Hamiltonian matrix and on `ALASKA` for computing the extra contributions to the gradient.

8.11.3 Files

Below is a list of the files that are used/created by the `ESPF` program.

Input files

File	*Contents*
ESPFINP	The ESPF input file.
SEWARINP	The Seward input file.
RYSRW	Data base for the fast evaluation of roots and weights of the Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
ABDATA	Data base for the evaluation of roots and weights of high order Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
ONEINT	One-electron integral file generated by the `SEWARD` program.
RUNFILE	File for communication of auxiliary information.

Please note that the external potential can be given within a file, separated from the `ESPF` input file.

Intermediate files

All the intermediate files are related to the use of `ESPF` together with a MM code (i.e. `TINKER`) and allow for communication between the two programs. *MOLCAS* uses one file to pass the QM atoms coordinates and ESPF-derived point charges to `TINKER`. `TINKER` uses the same file to pass the external potential, the MM-only energy and gradient components to *MOLCAS*.

File	*Contents*
TINKER.LOG	The log file of the Tinker run
$Project.xyz	The coordinates file for `TINKER`.
$Project.key	The keywords file for `TINKER`.
$Project.qmmm	The communication file between *MOLCAS* and `TINKER`.

Output files

File	*Contents*
ONEINT	One-electron integral file generated by the `SEWARD` program.
RUNFILE	Communication file for subsequent programs.
ESPF.DATA	Ascii file containing some specific informations needed for subsequent calls to the `ESPF` module.

8.11.4 Input

Below follows a description of the input to `ESPF`.

In addition to the keywords and the comment lines the input may contain blank lines. The input for each module is preceded by its name like:

```
␣&ESPF
```

The first four characters of the keywords are decoded while the rest are ignored. However, for a more transparent input we recommend the user to use the full keywords.

Compulsory keywords

Keyword	*Meaning*
END OF INPUT	This marks the end of the input to the program.
EXTERNAL	Specify how the external potential is given. This keyword is compulsory in the first run of `ESPF`. On the next line, three possibilities are allowed: • One integer n is given. If n is 0, the next lines give the numbering, the values for the external potential, the field and field gradients for each atom. If n is greater than 0, the n next lines specify the sources of the external potential, each line gives three cartesian coordinates, one point charge, and (optionally) three dipole components. If Å are used as the length unit, the ANGSTROM keyword must be given right after n.

- One word beginning with ”@”. Then the following characters up to the next space is taken as a file name and the rest of the line is ignored. Instead, all input (including the first line) is read from the specified file and must follow the syntax specified above. However, if this word is ”@TINKER”, it means that the current job is a QM/MM job using the MOLCAS/TINKER interface. Accordingly the external potential will be computed directly by TINKER. Note TINKER requires at least two input files, ending with .xyz (coordinates) and .key (keywords). These files must use the name of the current MOLCASproject. Optionally, you can add the ”Mulliken” keyword or the ”Loprop” keyword after ”@TINKER”: it indicates what kind of charges are passed to TINKER. These charges may be used during the MM microiterations. If no keyword is given, the ESPF multipoles are selected.
- The NONE keyword meaning that no external potential is given. Accordingly, the ESPF module will compute the point charges (and optionally dipoles) derived from the electrostatic potential due to all electrons and nuclei.

Optional keywords

Keyword	*Meaning*
Title	Title of the job.
MultipoleOrder	Give the multipolar order of the ESPF operators. Only 0 (charge) or 1 (charge and dipole) are allowed and should be written on the next input line. Default value is 0.
Grid	Modify the grid specifications. The grid is made of points belonging to molecular surfaces defined according to the van der Waals radii of each quantum atom. Two schemes are available. The first one is the GEPOL procedure, as implemented into the PCM SCRF method. The other one is called PNT and is the default. On the next line, first select the method with the GEPOL or PNT option. On the same line, one integer number and one real number are given if PNT is selected. The first one gives the maximum number of shells around the van der Waals surface of the quantum atoms. The second one gives the distance between the shells. Note that all points within the van der Waals envelope are discarded to avoid the penetration effects. Default values are 4 shells separated by 1 Å. Alternatively, if GEPOL is selected, the same line must contain 1 integer indicating the number of surfaces to be computed (must be ¡ 6).
Show	Requires the printing of the ESPF.DATA file.
LAMorokuma	Set on the Morokuma’s scheme for scaling the link atom positions (QM/MM calculation). Note the scaling factor is currently hard-coded and is actually determined from the radii of the atoms involved in the QM/MM frontier bond.

Below follows a typical input for the calculation of the energy and the gradient of glycine molecule feeling the external potential of 209 TIP3P water molecules.

For more advanced QM/MM input, especially about Molcas/Tinker, see:

`http://sites.univ-provence.fr/lcp-ct/ferre/nf_tinker_qmmm.html`.

```
␣␣&Seward
Title
␣␣Glycine(QM)+H20(MM)
Basis␣set
C.sto-3g.....
␣␣C1␣␣␣1.11820␣␣␣␣␣0.72542␣␣␣␣-2.75821␣Angstrom
␣␣C2␣␣␣1.20948␣␣␣␣␣0.66728␣␣␣␣-1.25125␣Angstrom
End␣of␣basis
Basis␣set
O.sto-3g.....
␣␣O1␣␣␣2.19794␣␣␣␣␣1.10343␣␣␣␣-0.67629␣Angstrom
End␣of␣basis
Basis␣set
H.sto-3g.....
␣␣H1␣␣␣2.02325␣␣␣␣␣1.18861␣␣␣␣-3.14886␣Angstrom
␣␣H2␣␣␣0.25129␣␣␣␣␣1.31794␣␣␣␣-3.04374␣Angstrom
␣␣H3␣␣␣1.02458␣␣␣␣-0.28460␣␣␣␣-3.15222␣Angstrom
End␣of␣basis
Basis␣set
N.sto-3g.....
␣␣N1␣␣␣0.17609␣␣␣␣␣0.12714␣␣␣␣-0.61129␣Angstrom
End␣of␣basis
Basis␣set
C.sto-3g.....
␣␣C3␣␣␣0.09389␣␣␣␣-0.01123␣␣␣␣␣0.84259␣Angstrom
␣␣C4␣␣-1.21244␣␣␣␣-0.67109␣␣␣␣␣1.28727␣Angstrom
End␣of␣basis
Basis␣set
O.sto-3g.....
␣␣O2␣␣-2.06502␣␣␣␣-1.02710␣␣␣␣␣0.48964␣Angstrom
End␣of␣basis
Basis␣set
H.sto-3g.....
␣␣H4␣␣-0.61006␣␣␣␣-0.21446␣␣␣␣-1.14521␣Angstrom
␣␣H5␣␣␣0.92981␣␣␣␣-0.61562␣␣␣␣␣1.19497␣Angstrom
␣␣H6␣␣␣0.16338␣␣␣␣␣0.97444␣␣␣␣␣1.30285␣Angstrom
End␣of␣basis
Basis␣set
N.sto-3g.....
␣␣N2␣␣-1.41884␣␣␣␣-0.85884␣␣␣␣␣2.57374␣Angstrom
End␣of␣basis
Basis␣set
H.sto-3g.....
␣␣H7␣␣-0.73630␣␣␣␣-0.57661␣␣␣␣␣3.25250␣Angstrom
␣␣H8␣␣-2.28943␣␣␣␣-1.29548␣␣␣␣␣2.82140␣Angstrom
End␣of␣basis
␣&espf
MultipoleOrder
␣␣0
External
␣␣0
1␣␣-0.048␣-0.002␣-0.006␣-0.001␣␣0.007␣-0.009␣␣0.002␣-0.001␣␣0.001␣-0.001
2␣␣-0.047␣-0.002␣␣0.001␣-0.002␣␣0.003␣␣0.000␣-0.004␣␣0.000␣-0.001␣␣0.000
3␣␣-0.053␣␣0.004␣␣0.000␣-0.011␣␣0.002␣␣0.002␣-0.004␣␣0.002␣␣0.003␣-0.007
4␣␣-0.046␣␣0.011␣-0.009␣-0.001␣␣0.006␣-0.005␣-0.001␣␣0.003␣␣0.003␣-0.004
5␣␣-0.042␣-0.016␣-0.011␣-0.006␣␣0.005␣-0.007␣␣0.003␣-0.004␣-0.001␣-0.005
6␣␣-0.050␣␣0.000␣␣0.008␣␣0.001␣␣0.006␣-0.006␣␣0.000␣-0.002␣␣0.000␣-0.001
7␣␣-0.039␣-0.008␣␣0.001␣␣0.000␣␣0.001␣-0.002␣␣0.001␣-0.001␣-0.001␣-0.001
8␣␣-0.032␣-0.007␣-0.002␣␣0.004␣␣0.002␣-0.003␣␣0.001␣-0.002␣␣0.002␣-0.001
```

```
9␣␣-0.011␣-0.009␣␣0.004␣␣0.001␣␣0.002␣␣0.000␣-0.002␣-0.001␣␣0.001␣␣0.001
10␣␣0.000␣-0.011␣␣0.003␣␣0.004␣␣0.001␣␣0.002␣-0.003␣␣0.001␣-0.001␣␣0.001
11␣-0.028␣-0.008␣␣0.004␣-0.001␣-0.001␣-0.002␣␣0.002␣-0.001␣␣0.001␣-0.002
12␣-0.026␣␣0.003␣-0.008␣␣0.014␣␣0.002␣-0.001␣-0.001␣-0.008␣␣0.006␣-0.009
13␣-0.037␣-0.008␣-0.003␣␣0.004␣-0.007␣␣0.007␣␣0.000␣␣0.001␣␣0.007␣-0.001
14␣-0.016␣-0.007␣␣0.007␣-0.008␣␣0.003␣␣0.003␣-0.006␣␣0.000␣␣0.002␣␣0.002
15␣-0.025␣␣0.003␣␣0.012␣-0.007␣␣0.003␣-0.001␣-0.002␣-0.006␣␣0.005␣␣0.009
16␣-0.010␣-0.011␣␣0.000␣-0.014␣␣0.001␣␣0.007␣-0.008␣␣0.001␣␣0.000␣-0.001
end␣of␣input
␣&scf
Title
␣␣SCF
Charge
␣␣0
end␣of␣input
␣&alaska
end␣of␣input
```

8.12 EXPBAS

The program EXPBAS is used to expand an orbital file to a larger basis set. It should only be used with generally contracted basis sets (ANO or CC). The original orbital file is ***INPORB***. The Runfile, ***RUNFIL1***, from the small calculation is also needed. One runs also the GATEWAY and SEWARD (with the keyword ONEONLY) for the larger basis set to obtain the corresponding runfile, ***RUNFIL2***. The program will produce the file ***EXPORB***, which can be used as the input orbital file for calculations using the larger basis set. No input is needed.

8.12.1 Dependencies

The EXPBAS depends on one InpOrb file and two Runfiles produced by other programs.

8.12.2 Files

Input files

Three input files are needed: ***INPORB***, an orbital file for the small basis set. ***RUNFIL1***, the runfile for the small basis set. ***RUNFIL2***, the runfile for the larger basis set.

Output files

EXPBAS generates the file ***EXPORB*** which contains orbitals, orbital energies, occupation numbers, and type indices for the larger basis set.

8.12.3 Input

There is no input to the EXPBAS except the name of the module:

```
␣&EXPBAS
```

Input example

```
>>COPY␣name.Inporb␣INPORB
>>COPY␣name.RunFil2␣RUNFIL1
>>COPY␣name.Runfil2␣RUNFIL2
␣&EXPBAS
```

Comments: This is the strategy:

1. Do a calculation (SCF, DFT, CASSCF, etc) with a small basis set. Save the produced orbital file and the runfile as: name.Inporb and name.Runfil1.
2. Run a `GATEWAY` to create the second runfile, which is then saved as name.Runfil2
3. Run `EXPBAS` as indicated above. The file name.ExpOrb is produced. This file can be used as input for large basis set calculations. Note that the inactive and active orbitals will have the same place in the list as with the smaller basis set, so they can be easily identified.

8.13 FFPT

The program `FFPT` prepares the one-electron integral file generated by `SEWARD` for subsequent finite-field perturbation calculations. To do so, the core Hamiltonian matrix is always reconstructed from the nuclear attraction and kinetic energy integrals. The perturbation matrix is then added to the core Hamiltonian matrix where the external perturbation and its strength is specified by input. Any suitable combination of the perturbations is allowed. Following some examples

1. **Dipole moment operator:** This option corresponds to a homogeneous external field perturbation and can be used to calculate dipole moments and dipole polarizabilities.
2. **Quadrupole and higher electric moment operators:** This option corresponds to a non homogeneous external field perturbation and can be used to calculate quadrupole moments and quadrupole polarizabilities, etc.
3. **Relativistic corrections:** This option is used to calculate perturbational relativistic corrections (sum of the mass-velocity and the one-electron Darwin contact term) to the total energy. Note that care must be taken to avoid variational collapse, i.e. the pertrubation correction should be small.

For a complete list of one-electron integrals which can be evaluated by the program `SEWARD` check out the section 8.33.1 and, especially, the subsection 8.33.1

Note, the perturbation matrices consist of the electronic contributions, only. The quadrupole, electric field gradient and higher electric moment perturbation matrices are given as the traceless tensors.

8.13.1 Dependencies

In order to complete successfully, the program `FFPT` needs the one-electron integral file. The latter must include all types of integrals needed to construct the perturbed one-electron Hamiltonian.

8.13.2 Files

Input files

The program `FFPT` needs the following files on input:

File	*Contents*
ONEINT	One-electron integral file produced by `SEWARD`. It is assumed to contain the matrix elements which are needed to construct the perturbation operator.

Output files

The program `FFPT` creates/updates the following files on output:

File	*Contents*
ONEINT	The one-electron integral file is modified by the program `FFPT`.

8.13.3 Input

The input to the `FFPT` program begins with the program name:

```
␣&FFPT
```

Similar to all programs in *MOLCAS* the input is given as a set of keywords and, if necessary, is followed by supplementary lines including the input data. The first four characters of the keywords are decoded while the rest are ignored.

General keywords

The following keywords are known to the `FFPT` utility:

Keyword	*Meaning*
TITLE	This command marks the beginning of the title and can be followed by at most 10 cards of input.

DIPO	Add the dipole moment perturbation operator. By default, the dipole moment integrals are always computed with respect to the center of nuclear charge. The keyword is followed by up to three additional input lines. Each line consists of two entries, the component of the dipole operator and the perturbation length. The component is specified by a single letter (X, Y or Z).
QUAD	Add the quadrupole moment perturbation operator. The keyword is followed by at least one additional input line and may be complemented by as many additional lines as needed. Each line consists of two entries, the component of the operator and the perturbation strength. The component is specified by a pair of letters (XX, XY, XZ, YY, YZ or ZZ). By default, the quadrupole moment integrals are calculated with respect to the center of mass. For any other selection the origin of the perturbation operator also needs to be specified by entering a line starting with the string ORIG followed by the coordinates.
OCTU	Add the octupole moment perturbation operator. The keyword is followed by at least one additional input line and may be complemented by as many additional lines as needed. Each line consists of two entries, the component of the operator and the perturbation strength. The component is specified by a triple of letters (XXX, XXY, XXZ, XYY, XYZ, XZZ, YYY, YYZ, YZZ, or ZZZ). By default, the octupole moment integrals are calculated with respect to the center of mass. For any other selection the origin of the perturbation operator also needs to be specified by entering a line starting with the string ORIG followed by the coordinates.
EFLD	Add the electric field perturbation operator. The keyword is followed by at least two additional input lines and may be complemented by as many additional lines as needed. Each line consists of two entries, the component of the operator and the perturbation strength. The component is specified by a single letter (X, Y or Z). In addition, the origin of the perturbation operator also needs to be specified by entering a line starting with the string ORIG followed by the coordinates.
EFGR	Add the electric field gradient perturbation operator. The keyword is followed by at least one additional input line and may be complemented by as many additional lines as needed. Each line consists of two entries, the component of the operator and the perturbation strength. The component is specified by a pair of letters (XX, XY, XZ, YY, YZ or ZZ). In addition, the origin of the perturbation operator also needs to be specified by entering a line starting with the string ORIG followed by the coordinates.
RELA	Add the relativistic correction (mass-velocity and one-electron Darwin contact term). The command is followed by one additional line of input specifying the perturbation strength.

GLBL This command marks the beginning of a more general perturbation description which is not included as a subcommand of the FFPT command. This card is followed by as many additional input lines as needed and is terminated if the next input line starts with a command. Each input line contains only one perturbation description and three data fields which are: Label, component and perturbation strength. The label consists of a character string of length 8 and names the one- electron integrals produced by `SEWARD`. The component of an operator is given as an integer. The last parameter denotes the strength of a perturbation operator and is given as a real number. For a list of the available one-electron integral labels refer to section 8.33.

For example to add Pauli repulsion integrals for reaction field calculations the input would look like:

```
␣&FFPT
GLBL
␣'Well␣␣␣1'␣1␣1.000
␣'Well␣␣␣2'␣1␣1.000
␣'Well␣␣␣3'␣1␣1.000
```

SELEctive With the same localization scheme as used in `LOPROP`, the perturbation from `FFPT` is localized in an orthogonal basis. Then the user can specify on which basis functions the perturbation should act. For example, the input

```
␣&FFPT
DIPO
X␣0.005
SELECTIVE
2
.true.␣1␣26
.false.␣67␣82
.true.
0.5
```

leads to that the perturbation only acts on densities with (1) both basis function indeces in the set $\{1, \cdots, 26\}$ or (2) one index in the set $\{1, \cdots, 26\}$ while the other is in the set $\{67, \cdots, 82\}$, and in this case the perturbation should be multiplied by 0.5.; all other densities are unaffected by the perturbation. We call the former type of subset an atom domain and the latter a bond domain. Generally, the input structure is this: First line specifies how many subsets, N, that will be defined. Then follow N lines starting with a logical flag telling if the subset is an atom domain with the starting and ending basis function indeces thereafter. $N-1$ lines follow where the bond domain is defined in the following way:

```
Do i=2,nSets
  Read(*,*)(Bonds(i,j),j=1,i-1)
Enddo
```

Finally a scalar is given which scales the defined bond domains.

The LoProp-functions will almost coincide with the original input AO-basis, although the localization will modify the meaning slightly,

	hence it is not possible to exactly localize the perturbation to a group of atoms; `LOPROP` is a way to come close to perfect localization. `FFPT` calls `LOPROP` internally and no call to `LOPROP` has to specified by the user.
CUMUlative	Adds the perturbation to the current H0, enabling many consecutive FFPT calls. Without this keyword, the perturbation always starts from the unperturbed H0.

Input example

The following input will prepare the one-electron integral file generated by `SEWARD` for subsequent finite-field perturbation calculations by adding a linear electric field in z-direction.

```
␣&FFPT
DIPO
Z␣0.001
```

Response properties are obtained by numerical differentiation of the total energy with respect to the field parameter. For definitions of the response properties the interested reader is referred to the paper of A.D. Buckingham in *Adv. Chem. Phys.*, Vol 12, p 107 (1967). According to the definition of the dipole moment, it is obtained as the first derivative of the energy with respect to the field strength. Similarly, the dipole polarizability is given by the second derivative of the energy with respect to the field strength.

8.14 GATEWAY

The Gateway module collects information about molecular system (geometry, basis sets, symmetry) to be used for future calculations.

Gateway module is a subset of `SEWARD`. All keywords for this module can also appear as an input for seward, however, for clearity the information about molecular system can be placed as an input for this module. Note, that gateway module does not compute any integral, and so must be followed by run of seward module (probably with a void input).

`GATEWAY` destroys the communication file *`RUNFILE`*, if it is used in a combination with geometry optimization it should run outside the optimization loop.

8.14.1 Input

This sections will describe the various possible input blocks in `GATEWAY`. These controll

- the molecular structure (coordinates, symmetry and basis sets),
- explicit auxiliary basis sets in terms of CD basis sets (aCD and acCD) or external auxiliary basis sets,
- parameters for reaction field calculations, i.e. parameters for the Kirkwood model or the PCM model and options for Pauli repulsion integral and external field integrals,

- options for finite nuclear charge distribution models in assocation with relativistic calculations, and
- the option to use the Saddle method to locate transitions state geometries.

The `GATEWAY` input section always starts with the program reference:

```
␣&GATEWAY
```

Molecular structure: coordinates, symmetry and basis sets

There are three different ways to specify the molecular structure, symmetry and the basis sets in `GATEWAY`:

- the so-called native input (old molcas standard),
- XYZ input and
- Z-matrix input.

The three different modes will be described below.

Native input If the geometry is specified in a native molcas format, only symmetry unequvalent atoms should be specified. The default units are atomic units. By default, symmetry is not used in the calculation.

Keyword	*Meaning*
SYMMETRY	Symmetry specification follows on next line. There may be up to three different point group generators specified on that line. The generators of a point group is the minimal set of symmetry operators which is needed to generate all symmetry operators of a specific point group. A generator is in the input represented as a sequence of up to three of the characters x, y, and z. The order within a given sequence is arbitrary and the generators can be given in any sequence. Observe that the order of the irreps is defined by the order of the generators as (E, g_1, g_2, g_1g_2, g_3, g_1g_3, g_2g_3, $g_1g_2g_3$)! Note that E is always assumed and should never be specified. Below is listed the possible generators. • **x** — Reflection in the yz-plane. • **y** — Reflection in the xz-plane. • **z** — Reflection in the xy-plane. • **xy** — Twofold rotation around the z-axis. • **xz** — Twofold rotation around the y-axis. • **yz** — Twofold rotation around the x-axis.

- **xyz** — Inversion through the origin.

The default is no symmetry.

BASIs Set

This notes the start of a basis set definition. The next line always contains a basis set label. For the definitions of basis set labels see the subsequent sections. Below follows a description of the options associated with the basis set definition.

- **Label** [/ **option**] - The label is a specification of a specific basis set, e.g. C.ANO...4s3p2d., which is an ANO basis set. If no option is specified `GATEWAY` will look for the basis set in the default basis directory. If an option is specified it could either be the name of an alternative basis directory or the wording "Inline" which defines that the basis set will follow in the current input file. For the format of the **Inline** option see the section 'Basis set format'. Observe that the label is arbitrary for this option and will not be decoded. The **Label** card is mandatory.
- **Name x, y, z (Angstrom or Bohr)** - This card specifies an arbitrary (see next sentence!) name for a symmetry distinct center and its Cartesian coordinates. Observe, that the name "DBAS" is restricted to assign the center of the diffuse basis functions required to model the continuum orbitals in R-matrix calculations. The label is truncated to four characters. Observe that this label must be unique to each center. The coordinate unit can be specified as an option. The default unit is Bohr. There should at least be one card of this type in a basis set definition.
- **Charge** - The real entry on the subsequent line defines the charge associated with this basis set. This will override the default which is defined in the basis set library. The option can be used to put in ghost orbitals as well as to augment the basis sets of the library. The **Charge** card is optional.
- **Spherical** (option) - Specifying which shells will be in real spherical Gaussians. Valid options are "all" or a list of the shell characters separated by a blank. The shell characters are s, p, d, f, etc. All shells after p are by default in real spherical Gaussians, exept for the d-functions in the 6-31G family of basis sets which are in Cartesian. The **Spherical** card is optional. The s and p shells and the d-functions of the 6-31G family of basis sets are by default in Cartesian Gaussians.
- **Contaminant** (option) - Specifying for which shells the contaminant will be kept. The contaminants are functions of lower rank which are generated when a Cartesian shell is transformed to a spherical representation (e.g. $r^2 = x^2 + y^2 + z^2$ for d-shells, p contaminants for f-shells, s and d contaminants for g-shells, etc). Valid options are the same as for the **Spherical** keyword. The default is no contaminant in any shell. The **Contaminant** card is optional.

- **End of Basis set** - Marks the end of the basis set specification. This card is mandatory.

Example of an input in native molcas format:

```
␣&GATEWAY
Title
formaldehyde
SYMMETRY
X␣Y
Basis␣set
H.STO-3G....
H1␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.924258␣␣␣-1.100293␣/Angstrom
End␣of␣basis
Basis␣set
C.STO-3G....
C3␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣-0.519589␣/Angstrom
End␣of␣basis
Basis␣set
O.STO-3G....
O␣␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣0.664765␣/Angstrom
End␣of␣basis
End␣of␣input
```

XYZ input If the geometry is specified in XYZ format, all atoms should be specified. The default units are Ångstroms. By default, maximum possible symmetry is used.

'Molcas XYZ' file format is an extension of plain XYZ format.

First line of this file contains the number of atoms.

Second line (a comment line) can contain 'a.u.' or 'bohr' to use atomic units, instead of default Ångstroms. Also this line can contain keyword TRANS, followed by 3 numbers, and/or ROTATE, followed by 9 numbers (in this case coordinates will be Translated by specified vector, and/or Rotated).

Remaining lines are used to specify Element and cartesian coordinates.

Element name might be optionally followed by Number, Label (separated by _ sign), or Basis Set (separated by .)

Examples:

```
H
H7
H_LABEL
H.STO-3G
H7_LABEL.STO-3G
```

Keyword	*Meaning*
TITLE	The keyword followed by a title.
COORD	The keyword (it can appear more than once) followed on the next line by the name of XYZ file, or inline coordinates in XYZ format. If the file is located in the same directory, where molcas job was submitted there is no need to specify the PATH to this file.

BASIS

The keyword can be used to specify global basis set for all atoms, or for a group of atoms. The keyword followed by a label of basis set, or by coma separated list of basis sets for individual atoms.

Example:

```
COORD
4
C          0.00000 0.00000 0.00000
H          1.00000 0.00000 0.00000
H          0.00000 1.00000 0.00000
H          0.00000 0.00000 1.00000
BASIS
STO-3G, H.6-31G*
```

In this example, the C atom (in the origin) will have the basis set STO-3G and the H atoms 6-31G*.

If keyword BASIS never appears in the input, the default basis, ANO-S-MB, will be used.

GROUP

The keyword can be used to specify the symmetry of the molecule.

The keyword must be followed by one of:

- FULL (default) - use maximum possible subgroup of D_{2h}
- NOSYM (same as E, or C1)
- space separated list of generators: e.g. X XY (for more details see SYMMETRY keyword)

Limitations: in the current implementation atom labels, and basis sets are ignored during symmetry recognition.

If XYZ input has been used in `GATEWAY`, a file with native molcas input will be produced and stored in working directory under the name ***`findsym.std`***.

Advanced keywords:

Keyword	*Meaning*
SYMTHRESHOLD	followed by a real number - threshold for symmetry recognition (default is 0.1)
MOVE	allow to translate and rotate molecule in order to find highest possible symmetry. (this is a default for all groups, except of C_1)
NOMOVE	do not allow to transform coordinates while searching for highest group (default for C_1 group)
BSSE	followed by an integer. Indicates which xyz-file that should be treated like ghost atoms.
RTRN	Max number of atoms for which bond lengths, angles and dihedral angles are listed, and the radius defining the maximum length of a bond follows on the next line. The latter is used as a threshold when printing out angles and dihedral angles. The length can be followed

by Bohr or Angstrom which indicates the unit in which the length was specified, the default is Bohr. The default values are 15 and 3.0 au.

```
Example:
 &GATEWAY
COORD
water.xyz
BASIS
STO-3G
```

or, in short EMIL notation:

```
 &GATEWAY
COORD=water.xyz; BASIS=STO-3G
```

Z-matrix input

Keyword	*Meaning*
ZMAT	Alternative format to give basis set and coordinates in terms of bond lengths, bond angles, and dihedral angles. First the basis set for all atoms (one time only) must be give in the same format as **Label** in BASIs keyword. Only the basis sets in the MOLCAS library are allowed. **End of Basis set** or a blank line mark the end of the basis set specification. The geometry as Z-matrix format follows. Each line of a Z-matrix gives the internal coordinates for one of the atoms within the molecule with the following sintax: **Name I bond-length J bond-angle K dihedral-angle** **Name** is label (atomic symbol + string) for a symmetry distinct center L; **I bond-length** distance of L from atom I; **J bond-angle** planar angle between atoms L-I-J; **K dihedral-angle** dihedral angle between atoms L-I-J-K. Note that the first atom only requires the **Name** and defines the origin of cartesian axes. Second atom requires **Name I bond-length** and it will defines the Z axis. Third atom requires **Name I bond-length J bond-angle** and defines the XZ plane (and implicitly, the Y axis). Only numerical values must be used (no variable names) and Angstroms and degree are assumed as units. Two type of special atoms are allowed: *dummy* **X** atoms and *ghost* **Z** atoms. The former will appear in the calculations and they have a nuclear charge of 0. and have not electrons and Basis Set. They will also appear in the definition of internal coordinates in `SLAPAF`. The latter are used only within the Z-Matrix definition of the geometry but they will appear in the final Z-matrix section in `SLAPAF`. Both special atoms can be used to define the cartesian axes and the symmetry elements. **End of ZMAT** or a blank line mark the end of the section. Here is an example for (S)-1-chloroethanol (C_1 symmetry):

```
ZMAT
H.ANO-L...2s1p.
C.ANO-L...3s2p1d.
O.ANO-L...3s2p1d.
Cl.ECP.Huzinaga.7s7p1d.1s2p1d.7e-NR-AIMP.
End␣of␣basis
C1
O2␣␣␣␣␣␣1␣␣␣1.40000
C3␣␣␣␣␣␣1␣␣␣1.45000␣␣␣2␣␣␣109.471
H4␣␣␣␣␣␣1␣␣␣1.08900␣␣␣2␣␣␣109.471␣␣␣␣␣3␣␣␣120.000
Cl5␣␣␣␣␣1␣␣␣1.75000␣␣␣2␣␣␣109.471␣␣␣␣␣3␣␣-120.000
H6␣␣␣␣␣␣2␣␣␣0.94700␣␣␣1␣␣␣109.471␣␣␣␣␣3␣␣␣180.000
H7␣␣␣␣␣␣3␣␣␣1.08900␣␣␣1␣␣␣109.471␣␣␣␣␣2␣␣␣180.000
H8␣␣␣␣␣␣3␣␣␣1.08900␣␣␣1␣␣␣109.471␣␣␣␣␣7␣␣␣120.000
H9␣␣␣␣␣␣3␣␣␣1.08900␣␣␣1␣␣␣109.471␣␣␣␣␣7␣␣␣240.000
End␣of␣z-matrix
```

In geometry optimization, `SLAPAF` will regenerate the coordinates as Z-matrix in the section with the summary concerning each iterations. This will be possible only if *ghost* atoms are used within the first three atoms or if they are not used at all.
Both BASIs and ZMAT cards can be used at the same time. Here is an example for a complex between methanol and water (C_s symmetry):

```
Symmetry
␣Y
ZMAT
H.ANO-L...1s.
C.ANO-L...2s1p.
O.ANO-L...2s1p.
End␣of␣basis
C1
O2␣␣1␣1.3350
H3␣␣1␣1.0890␣␣2␣109.471
H4␣␣1␣1.0890␣␣2␣109.471␣␣3␣-120.
H6␣␣2␣1.0890␣␣1␣109.471␣␣3␣␣180.
End␣of␣z-matrix
Basis␣set
O.ANO-L...2s1p.
␣O␣␣␣␣-2.828427␣␣␣␣␣0.000000␣␣␣␣␣2.335000␣␣/␣Angstrom
End␣of␣basis
Basis␣set
H.ANO-L...1s.
␣H␣␣␣␣-2.748759␣␣␣␣␣0.819593␣␣␣␣␣2.808729␣␣/␣Angstrom
End␣of␣basis
```

In this case `SLAPAF` will not regenerate the Z-matrix.

Explicit auxiliary basis sets

The so-called Resolution of Identity (RI) technique (also called Density Fitting, DF) is implemented in the MOLCAS package. This option involves the use of an auxiliary basis set in the effective computation of the 2-electron integrals. MOLCAS incorporates both the use of conventionally computed, externally provided, auxiliary basis sets (RIJ, RIJK, and RIC types), and on-the-fly generated auxiliary basis sets. The latter are atomic CD (aCD) or the atomic compact CD (acCD) basis sets, based on the Cholesky decomposition method. The externally provided auxiliary basis sets are very compact, since they are tailored for special wave function methods. However, they are not provided for all available valence basis sets. The aCD or acCD RI auxiliary basis sets are a more general option and provides auxiliary basis sets for any wave function model and valence basis set.

Keyword	*Meaning*
RIJ	Use the RI-J basis in the density fitting (DF) approach to treat the two-electron integrals. Note that the valence basis set must have a supporting auxiliary basis set for this to work.
RIJK	Use the RI-JK auxiliary basis in the density fitting (DF) approach to treat the two-electron integrals. Note that the valence basis set must have a supporting auxiliary basis set for this to work.
RIC	Use the RI-C auxiliary basis in the density fitting (DF) approach to treat the two-electron integrals. Note that the valence basis set must have a supporting auxiliary basis set for this to work.
RICD	Use the aCD or acCD approach [7] to treat the two-electron integrals. This procedure will use an on-the-fly generated auxiliary basis set.
CDTHRESHOLD	Threshold for on-the-fly generation of aCD or acCD auxiliary basis sets for RI calculations (default value 1.0d-4).
SHAC	Skip high angular combinations a la Turbomole when creating on-the-fly basis sets (default of).
KHAC	Keep high angular combinations when creating on-the-fly basis sets (default on).
ACD BASIS	Generate an atomic CD (aCD) auxiliary basis sets (default off).
ACCD BASIS	Generate an atomic compact CD (acCD) auxiliary basis sets (default on).

Reaction field calculations

The effect of the solvent on the quantum chemical calculations has been introduced in MOLCAS through the reaction field created by the surrounding environment, represented by a polarizable dielectric continuum outside the boundaries of a cavity containing the solute molecule. MOLCAS-4 support Self Consistent Reaction Field (SCRF) and Multi Configurational Self Consistent Reaction Field (MCSCRF) calculations within the framework of the `SCF` and the `RASSCF` programs. The reaction field, computed in a self-consistent fashion, can be later added as a constant perturbation for the remaining programs, as for example `CASPT2`.

The purpose of this facility is to incorporate the effect of the environment (a solvent or a solid matrix) on the studied molecule. The utility itself it is not a program, but requires an additional input which has to be provided to the `GATEWAY` program. Two methods are available for SCRF calculations: one is based on the Kirkwood model, the other is the so called Polarizable Continuum Model (PCM). The reaction field is computed as the response of a dielectric medium polarized by the solute molecule: the solute is placed in a “cavity” surrounded by the dielectric. In Kirkwood model the cavity is always spherical, whereas in PCM the cavity is modeled on the actual solute shape.

The possible set of parameters controlled by input are:

- the Kirkwood model,
- the PCM model, and
- one-elctron integrals representing Pauli repulsion and external fields.

First a brief presentation of the Kirkwood and the PCM models.

The Kirkwood Model The Kirkwood model is an expansion of the so-called Onsager model where the surrounding will be characterized by its dielectric permitivity and a radius describing a spherical cavity, indicating where the dielectric medium starts. (Note that all atoms in the studied molecule must be inside the spherical cavity.) The Pauli repulsion due to the medium can be introduced by use of the spherical well integrals which are generated by `SEWARD`. The charge distribution of the molecule will introduce an electric field acting on the dielectric medium. This reaction field will interact with the charge distribution of the molecule. This interaction will manifest itself as a perturbation to the one-electron Hamiltonian. The perturbation will be automatically computed in a direct fashion (no multipole integrals are stored on disk) and added to the one-electron Hamiltonian. Due to the direct way in which this contribution is computed rather high terms in the mutipole expansion of the charge can be afforded.

The Polarizable Continuum Model, PCM The PCM has been developed in order to describe the solvent reaction field in a more realistic way, basically through the use of cavities of general shape, modeled on the solute. The cavity is built as the envelope of spheres centered on solute atoms or atomic groups (usually, hydrogen atoms are included in the same sphere as the heavy atoms they are bonded to). The reaction field is described by means of apparent charges (solvation charges) spread on the cavity surface, designed to reproduced the electrostatic potential due to the polarized dielectric inside the cavity. Such charges are used both to compute solute-solvent interactions (modifying the total energy of the solute), and to perturb the molecular Hamiltonian through a suitable operator (thus distorcing the solute wave-function, and affecting all the electronic properties). The PCM operator contains both one- and two-electron terms: it is computed using atomic integrals already present in the program, through a “geometry matrix” connecting different points lying on the cavity surface. It can be shown that with this approach the SCF and RASSCF variational procedures lead to the free energy of the given molecule in solution: this is the thermodynamic meaning of the SCF or CI energy provided by the program. More precisely, this is the solute-solvent electrostatic contribution to the free energy (of course, other terms depending on solute atomic motions, like vibrational and rotational free energies, should be included separately); it can be used to get a good approximation of the solvation free energy, by subtracting the SCF or CI energy computed in vacuo, and also to compute directly energy surfaces and reaction paths in solution. On the other hand, the solute wave-function perturbed by the reaction field can be used to compute any electronic property in solution.

Also other quantities can be computed, namely the cavitation free energy (due the the work spent to create the cavity in the dielectric) and the dispersion-repulsion free energy: these terms affect only the total free energy of the molecule, and not its electronic distribution. They are collectively referred to as non-electrostatic contributions.

Note that two other keywords are defined for the RASSCF program: they refer to the CI root selected for the calculation of the reaction field (RFROOT), and to the possibility to perform a non-equilibrium calculation (NONEQ) when vertical electronic transitions are studied in solution. These keywords are referenced in the RASSCF section. To include the reaction field perturbation in a SCF, RASSCF or CASPT2 calculation, another keyword must be specified (RFPERT), as explained in the respective program sections.

Complete and detailed examples of how to add a reaction field, through the Kirkwood or the PCM model, into quantum chemical calculations in *MOLCAS* is presented in section 5.6 of the examples manual. The user is encouraged to read that section for further details.

Input for the Kirkwood and PCM models

Files The reaction field calculations will store the information in the following files, which will be used by the following programs

File	*Contents*
ONEINT	One-electron integral file used to store the Pauli repulsion integrals
RUNFILE	Communications file. The last computed self-consistent reaction field (SCF or RASSCF) will be stored here to be used by following programs
GV.off	Input file for the external program “geomview” (see Tutorial section “Solvent models”), for the visualization of PCM cavities

Input Below follows a description of the input to the reaction field utility in the GATEWAY program. The RASSCF program has its own keywords to compute reaction fields for excited states.

In addition to the keywords and the comment lines the input may contain blank lines. The input is always preceded by the following keyword in the SEWARD input

RF-Input

The first four character of the keywords are decoded while the rest are ignored. However, for a more transparent input we recommend the user to use the full keywords.

Compulsory keywords

Keyword	*Meaning*

END Of RF-Input	This markes the end of the input to the reaction field utility.

Optional keywords for the Kirkwood Model

Keyword	*Meaning*
REACtion Field	This command is exclusive to the Kirkwood model. It indicates the beginning of the specification of the reaction field parameters. The subsequent line will contain the dielectric constant of the medium, the radius of the cavity in Bohrs (the cavity is always centered around the origin), and the angular quantum number of the highest multipole moment used in the expansion of the change distribution of the molecule (only charge is specified as 0, charge and dipole moments as 1, etc.). The input specified below specifies that a dielectric permitivity of 80.0 is used, that the cavity radius is 14.00 a.u., and that the expansion of the charge distribution is truncated after l=4, i.e hexadecapole moments are the last moments included in the expansion. Optionally a fourth argument can be added giving the value of the dielectric constant of the fast component of the solvent (default value 1.0).

Sample input for the reaction field part (Kirkwood model)

```
␣RF-Input
Reaction␣field
80.0␣14.00␣4
End␣Of␣RF-Input
```

Sample input for a complete reaction field calculation using the Kirkwood model. The `SCF` computes the reaction field in a self consistent manner while the `MRCI` program adds the effect as a constant perturbation.

```
␣&GATEWAY
Title
␣HF␣molecule
Symmetry
X␣Y
Basis␣set
F.ANO-S...3S2P.
F␣␣␣␣␣␣0.00000␣␣␣0.00000␣␣␣1.73300
End␣of␣basis
Basis␣set
H.ANO-S...2S.
H␣␣␣␣␣␣0.00000␣␣␣0.00000␣␣␣0.00000
End␣of␣basis
Well␣integrals
␣4
␣1.0␣5.0␣␣6.75
␣1.0␣3.5␣␣7.75
␣1.0␣2.0␣␣9.75
␣1.0␣1.4␣11.75
RF-Input
Reaction␣field
␣80.0␣4.75␣4
End␣of␣RF-Input
␣&SEWARD
```

```
␣&SCF
Title
␣HF␣molecule
Occupied
␣3␣1␣1␣0
␣&MOTRA
Title
␣HF␣molecule
LumOrb
Frozen
␣1␣0␣0␣0
RFPert
␣&GUGA
Title
␣HF␣molecule
Electrons
␣␣␣␣8
Spin
␣␣␣␣1
Symmetry
␣␣␣␣4
Inactive
␣␣␣␣2␣␣␣␣1␣␣␣␣1␣␣␣␣0
Active
␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0
CiAll
␣␣␣␣1
End␣of␣Input
␣&MRCI
Title
␣HF␣molecule
SDCI
```

Optional keywords for the PCM Model

Keyword	*Meaning*
PCM-MODEL	If no other keywords are specified, the program will execute a standard PCM calculation with water as solvent. The solvent reaction field will be included in all the programs (`SCF`, `RASSCF`, `CASPT2`, etc) invoked after `SEWARD`: note that in some cases additional keywords are required in the corresponding program sections. Some PCM parameters can be changed through the following keywords.
SOLVENT	Used to indicate which solvent is to be simulated. The name of the requested solvent must be written in the line below this keyword. Find implemented solvents in the PCM model below this section.
DIELECTRIC CONSTANT	Defines a different dielectric constant for the selected solvent; useful to describe the system at temperatures other that 298 K, or to mimic solvent mixtures. The value is read in the line below the keyword. An optional second value might be added on the same line which defines a different value for the infinite frequency dielectric constant for the selected solvent (this is used in non-equilibrium calculations; by default it is defined for each solvent at 298 K).
CONDUCTOR VERSION	It requires a PCM calculation where the solvent is represented as a polarized conductor: this is an approximation to the dielectric model which works very well for polar solvents (i. e. dielectric

constant greater than about 5), and it has some computational advantages being based on simpler equations. It can be useful in cases when the dielectric model shows some convergence problems.

AAREa	It is used to define the average area (in Å^2) of the small elements on the cavity surface where solvation charges are placed; when larger elements are chosen, less charges are defined, what speeds up the calculation but risks to worsen the results. The default value is 0.4 Å^2 (i. e. 60 charges on a sphere of radius 2 Å). The value is read in the line below the keyword.
R-MIn	It sets the minimum radius (in Å) of the spheres that the program adds to the atomic spheres in order to smooth the cavity surface (default 2 Å). For large solute, if the programs complains that too many sphere are being created, or if computational times become too high, it can be useful to enlarge this value (for example to 1 or 1.5 Å), thus reducing the number of added spheres. The value is read in the line below the keyword.
PAULing	It invokes the use of Pauling's radii to build the solute cavity: in this case, hydrogens get their own sphere (radius 1.2 Å).
SPHEre radius	It is used to provide sphere radii from input: for each sphere given explicitly by the user, the keyword "Sphere radius" is required, followed by a line containing two numbers: an integer indicating the atom where the sphere has to be centered, and a real indicating its radius (in Å). For example, "Sphere radius" followed by "3 1.5" indicates that a sphere of radius 1.5 Å is placed around atom #3; "Sphere radius" followed by "4 2.0" indicates that another sphere of radius 2 Å is placed around atom #4 and so on.

Solvents implemented in the PCM model are

Name	Dielectric constant	Name	Dielectric constant	Name	Dielectric constant
water	78.39	dichloroethane	10.36	toluene	2.38
dimethylsulfoxide	46.70	quinoline	9.03	benzene	2.25
nitromethane	38.20	methylenchloride	8.93	carbontetrachloride	2.23
acetonitrile	36.64	tetrahydrofuran	7.58	cyclohexane	2.02
methanol	32.63	aniline	6.89	heptane	1.92
ethanol	24.55	chlorobenzene	5.62	xenon	1.71
acetone	20.70	chloroform	4.90	krypton	1.52
isoquinoline	10.43	ethylether	4.34	argon	1.43

Sample input for the reaction field part (PCM model): the solvent is water, a surface element average area of 0.2 Å^2 is requested.

```
RF-input
```

```
PCM-model
Solvent
water
AAre
0.2
end␣of␣rf-input
```

Sample input for a standard PCM calculation in water. The `SCF` and `RASSCF` programs compute the reaction field self consistently and add its contribution to the Hamiltonian. The `RASSCF` is repeated twice: first the ground state is determined, then a non-equilibrium calculation on the first excited state is performed.

```
␣&SEWARD
Title
formaldehyde
Basis␣set
H.STO-3G....
H1␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.924258␣␣␣-1.100293␣/Angstrom
H2␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣-0.924258␣␣␣-1.100293␣/Angstrom
End␣of␣basis
Basis␣set
C.STO-3G....
C3␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣-0.519589␣/Angstrom
End␣of␣basis
Basis␣set
O.STO-3G....
O␣␣␣␣␣␣␣␣␣␣␣␣0.000000␣␣␣␣0.000000␣␣␣␣0.664765␣/Angstrom
End␣of␣basis
RF-input
PCM-model
solvent
water
end␣of␣rf-input
␣&SEWARD
␣&SCF
Title
formaldehyde
ITERATIONS
50
Occupied
8
␣&RASSCF
Title
h2co
nActEl
4␣0␣0
Symmetry
1
Inactive
6
Ras1
0
Ras2
3
Ras3
0
CiRoot
1␣1
1
Iter
100,20
LumOrb
␣&RASSCF
Title
```

```
h2co
nActEl
4␣0␣0
Symmetry
1
Inactive
6
Ras1
0
Ras2
3
Ras3
0
CiRoot
2␣2
1␣2
0␣1
Iter
100,20
JOBIPH
NonEq
RFRoot
2
```

Again the user is recommended to read section 5.6 of the examples manual for further details.

Keywords associated to one-electron integrals

Keyword	*Meaning*
WELL integrals	Request computation of Pauli repulsion integrals for dielectric cavity reaction field calculations. The first line specifies the total number of primitive well integrals in the repulsion integral. Then follows a number of lines, one for each well integral, specifying the coefficient of the well integral in the linear combination of the well integrals which defines the repulsion integral, the exponent of the well integral, and the distance of the center of the Gaussian from the origin. In total three entries on each line. All entries in atomic units. If zero or a negative number is specified for the number of well integrals a standard set of 3 integrals with their position adjusted for the radius of the cavity will be used. If the distance of the center of the Gaussian from the origin is negative displacements relative to the cavity radius is assumed.
XFIEld integrals	Request the presence of an external electric field represented by a number of partial charges and dipoles. Optionally, polarisabilities may be specified whose induced dipoles are determined self-consistently during the SCF iteration. The first line contains the total number of symmetry unique centers and optionally the word "Angstrom" to indicate that the coordinates are in units of angstrom. If there is a @ on the first line the following characters up to the next space is taken as a file name and the rest of the line is ignored. Instead, all input (including the first line) is read from the specified file. The first line may contain, apart from the first integer [nXF]

	(number of centers), up to four additional integers. The second integer [nOrd] specifies the maximum multipole order, or -1 signifying no permanent multipoles. Default is 1 (charges and dipoles). The third integer [p] specifies the type of external polarisabilities: 0 (default) no polarisabilities, 1 (isotropic), or 2 (anisotropic). The fourth integer [nFrag] specifies the number of fragments one multipole may contribute to (relevant only if polarisabilities are present). The default is 0, meaning that each permanent multipole is only excluded in the calculation of the field at its own polarisability, 1 means that one gives a fragment number to each multipole and that the static multipoles do not contribute to the polarising field within the same fragment, whereas 2 can be used in more complex situations, e.g. polymers, allowing you to specify a second fragment number so that junction atoms does not contribute to either of the neighbouring fragments. Finally, the fifth and last integer [nRead] (relevant only if Langevin dipoles are used) may be 0 or 1 (where 0 is default), specifying wheather an element number (e.g. 8 for oxygen) should be read for each multipole. In that case the default radius for that element is used to determine which Langevin grid points should be annihilated. A negative element number signifies that a particular radius should be used for that multipole, in thousands of a Bohr (-1400 meaning 1.4 Bohr). Then follows nXF lines, one for each center. On each line is first nFrag+nRead (which may equal 0) integers, specifying the fragments that the multipole should not contribute to (the first fragment is taken as the fragment that the polarisability belongs to) and the element number. Then follows the three coordinates of the center, followed by the multipoles and polarisabilities. The number of multipole entries is 0 for nOrd=-1, 1 for nOrd=0, 4 for nOrd=1, and 10 for nOrd=2. The number of polarisability entries are 0 for p=0, 1 for p=1, and 6 for p=2. The order of quadrupole moment and anisotropic polarisability entries is xx, xy, xz, yy, yz, zz. If default is used, i.e. only specifying the number of centers on the first line, each of these lines will contain 7 entries (coordinates, charge, and dipole vector). All entries are in atomic units, if not otherwise requested. To redirect input from external file the first line should have the form @*filename*.
RF-Input	Specification of reaction field parameters, consult the reaction field section of this manual.

Keywords associated with nuclear charge distribition models Input parameters associated with different models of the nuclear charge distribution. The default is to use a charge point representation.

Keyword	*Meaning*
FINIte	Request a finite center representation of the nuclei by a single exponent s-type Gaussian.

MGAUSSIAN	Request a finite center representation of the nuclei by a modified Gaussian.

The Saddle method for transition state optimization

The Saddle method [100] is a method to locate transition states (TS). The method, in practice, can be viewed as a series of constrained optimization along the reaction path, which connects two starting structure (could be the reactants and products of a reaction), to locate the region of the TS and a subsequent unconstrained optimization to locate the TS. The only data needed for the procedure are the energies and coordinates of the two structures. **Note** that this option will overwrite the coordinates which have already been specified with the normal input of the molecular geometry. However, this does not make that input section redundant and should always be included.

Keyword	*Meaning*
RP-COORDINATES	This activates the Saddle method for TS geometry optimization. The line is followed by an integer specifying the number of symmetry unique coordinates to be specified. This is followed by two sets of input - one line with the energy and then the Cartesian coordinates in bohr - for each of the two starting structures of the Saddle method. Note that the order of the coordinates must always match the order specified with the converntional input of the coordinates of the molecular system.

8.15 GENANO

`GENANO` is a program for determining the contraction coefficients for generally contracted basis sets [101]. They are determined by diagonalizing a density matrix, using the eigenvectors (natural orbitals) as the contraction coefficients, resulting in basis sets of the ANO (Atomic Natural Orbitals) type [102].

Some elementary theory: We can do a spectral resolution of a density matrix D

$$D = \sum_k \eta_k c_k c_k^\dagger \tag{8.3}$$

where η_k is the k'th eigenvalue (occupation value) and c_k is the k'th eigenvector (natural orbital). The occupation number for a natural orbital is a measure of how much this orbital contributes to the total one-electron density. A natural choice is to disregard the natural orbitals with small occupation numbers and use those with large occupation numbers to form contracted basis functions as

$$\varphi_k = \sum_i c_{ki} \chi_i \tag{8.4}$$

where χ_i is the i'th primitive basis function.

As a generalization to this approach we can average over density matrices from several wave functions, resulting in basis sets of the density matrix averaged ANO type, see for example

[103, 104, 105, 106]. We can view the averaging of density matrices as a sequence of rank-1 updates in the same way as in equation 8.3. We have more update vectors than the rank of the matrix, but this does not really change anything. The important observation is that all η's are positive and no information is lost in the averaging.

The general guideline for which wave functions to include is based on what you want to be able to describe. All wave functions you want an accurate description of should be included in the averaging.

As an example, let us consider the oxygen atom. We want to be able to describe the atom by itself accurately, thus a wave function for the atom is needed, usually at the CI level. In molecular systems, oxygen usually has a negative charge, thus including O^- is almost mandatory. A basis set derived from these two wave function is well balanced for the majority of systems containing oxygen. A logical conclusion would be that you need to include a few *molecular* wave functions of systems containing oxygen, but in practice this is not necessary. This is due to the fact that the degrees of freedom describing the orbital shape distortion when forming bonds are virtually identical to the lowest correlating orbitals. On the other hand, a few molecular species have oxygen with positive charge, thus it may be appropriate to include O^+ in the basis set.

A wide range of specialized basis sets can also be generated, for example a molecular basis set describing Rydberg orbitals, see the example in the "Tutorials and Examples" part, section 5.5.1. There is a possibility to create rydberg orbitals automatically by using the keyword RYDBERG. Here all unoccupied orbitals with negative orbital energies will be used with the associated occupation numbers

$$\eta_k = e^{6.9\times(\epsilon_k/\epsilon_0 - 1)} \tag{8.5}$$

where ϵ_k is the orbital energy of orbital k and ϵ_0 is the lowest orbital energy of all virtual orbitals. In order to use this option you need to use the `SCF` or `RASSCF` program to compute the orbitals for a cationic system.

You need one or more wave functions, represented by formatted orbital files, to generate the average density matrix. These natural orbital files can be produced by any of the wave function generators `SCF`, `RASSCF`, `MRCI` or `CPF`. You could also use `MBPT2` or `CASPT2`. This approach has been used in the generation of the ANO-RCC basis sets. Your specific requirements dictate the choice of wave function generator, but `MRCI` would be most commonly used.

You are not restricted to atomic calculations but can mix molecular and atomic calculations freely. The restrictions are that the name of the center, for which you are constructing a basis set, must be the same in all wave functions. The center may not be "degenerate", i.e. it may not generate other centers through symmetry operations. See the description of `SEWARD` on page 407 for a more extensive discussion. For example for O_2 you cannot use D_{2h} symmetry since this would involve one center that is mirrored into the other. Another restriction is, of course, that you must use the same primitive set in all calculations.

8.15.1 Dependencies

`GENANO` needs one or more wave functions in the form of natural orbitals. Thus you need to run one or more of `SCF`, `RASSCF`, `MRCI` or `CPF`. You could also use, for example, `MBPT2` or `CASPT2` but this is in general not recommended. `GENANO` also needs the one electron file *`ONEINT`* and the *`RUNFILE`* generated by `SEWARD`.

8.15.2 Files

Below is a list of the files that `GENANO` reads/writes. Files *ONEnnn*, *RUNnnn* and *NATnnn* must be supplied to the program. Files *ANO* and *FIG* are generated. File *PROJ* is an optional input file.

Input files

File	*Contents*
RUNnnn	This file contains miscellaneous information for the nnn'th wave function, generated by the program `SEWARD`. One file per wave function must be supplied, *RUN001*, *RUN002*, ...
ONEnnn	This file contains the one-electron integrals corresponding to the nnn'th wave function, generated by the program `SEWARD`. One file per wave function must be supplied, *ONE001*, *ONE002*, ...
NATnnn	This file contains the natural orbitals corresponding to the nnn'th wave function, generated by the appropriate wave function generating program. One file per wave function must be supplied, *NAT001*, *NAT002*, ...
PROJ	This file contains orbitals used for projection of the densities. Needs to be available if the keyword PROJECT is specified. It is compatible in format with the file *ANO*, and can thus be the the file *ANO* from a previous run of `GENANO`.

Output files

File	*Contents*
FIG	This file contains a PostScript figure file of eigenvalues.
ANO	This file contains the contraction coefficient matrix organized such that each column correspond to one contracted basis function.

8.15.3 Input

The input file must contain the line

```
␣&GENANO
```

right before the actual input starts. Below is a list of the available keywords. Please note that you can not abbreviate any keyword.

Keyword	*Meaning*
TITLE	This keyword starts the reading of title lines, with no limit on the number of title lines. Reading the input as title lines is stopped as soon an the input parser detects one of the other keywords. This keyword is *optional*.
SETS	This keyword indicates that the next line of input contains the number of sets to be used in the averaging procedure. This keyword must precede WEIGHTS if both are supplied. This keyword is *optional*, with one set as the default.
CENTER	This keyword is followed, on the next line, by the atom label for which the basis set is to be generated. The label must match the label you supplied to `SEWARD`. In previous versions of `GENANO` this label had to be in uppercase, but this restriction is now lifted and the case does not matter. This keyword is *compulsory*.
ROWWISE	This keyword makes `GENANO` produce the contraction coefficients rowwise instead of column-wise as is the default. This keyword is *optional*.
WEIGHTS	This keyword must be subsequent to keyword SETS if both are supplied. This keyword is *optional*, with equal weight on each of the sets as default.
PROJECT	This keyword states that you want to project out certain degrees of freedom from the density matrix. This can be useful for generating, for example, node less valence orbitals to be used with ECP's. If this keyword is specified, you must supply the file *`PROJ`* obtained as file *`ANO`* from a previous `GENANO` calculation, for instance. This keyword is *optional*.
LIFTDEGENERACY	This keyword will modify the occupation numbers read from the orbitals files. The purpose is to lift the degeneracy of core orbitals to avoid rotations. The occupation numbers are changed according to $\eta = \eta \times (1 + 10^{-3}/n)$ where n is the sequence number of the orbital in its irreducible representation. This keyword is *optional*.
RYDBERG	This keyword enables automatic generation of rydberg orbitals. With this keyword all occupied orbitals will get occupation number zero while the virtual orbitals will get a small occupation number decreasing with orbital number. Useful with a calculation on an cation where the virtual orbitals are near perfect rydberg orbitals. Note that you must use orbitals from the `SCF` or `RASSCF` program. This keyword is *optional*.
NOTHRESHOLD	This keyword is used to specify the threshold for keeping NO's (natural orbitals). Orbitals with occupation numbers less than the threshold are discarded. The threshold is read from the line following the keyword. Default value is 1.0d-8.

Below is a simple input example, where we construct an ANO basis set for the carbon atom. Two wave functions are used, the SCF wave function and the SDCI wave function for the ground state of the atom.

```
␣&SEWARD
Title
␣Carbon␣atom
Symmetry
x␣y␣z
Basis␣set
C.....␣/␣inline
␣␣6.0␣2
␣␣␣10␣␣␣10
5240.6353␣782.20479␣178.35083␣50.815942␣16.823562␣6.1757760␣2.4180490
.51190000␣.15659000␣.05480600
1.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.
0.␣1.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.
0.␣0.␣1.␣0.␣0.␣0.␣0.␣0.␣0.␣0.
0.␣0.␣0.␣1.␣0.␣0.␣0.␣0.␣0.␣0.
0.␣0.␣0.␣0.␣1.␣0.␣0.␣0.␣0.␣0.
0.␣0.␣0.␣0.␣0.␣1.␣0.␣0.␣0.␣0.
0.␣0.␣0.␣0.␣0.␣0.␣1.␣0.␣0.␣0.
0.␣0.␣0.␣0.␣0.␣0.␣0.␣1.␣0.␣0.
0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣1.␣0.
0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣0.␣1.
␣␣␣␣6␣␣␣␣6
18.841800␣4.1592400␣1.2067100␣.38554000␣.12194000␣.04267900
1.␣0.␣0.␣0.␣0.␣0.
0.␣1.␣0.␣0.␣0.␣0.
0.␣0.␣1.␣0.␣0.␣0.
0.␣0.␣0.␣1.␣0.␣0.
0.␣0.␣0.␣0.␣1.␣0.
0.␣0.␣0.␣0.␣0.␣1.
␣␣␣␣3␣␣␣␣3
1.2838000␣.34400000␣.09220000
1.␣0.␣0.
0.␣1.␣0.
0.␣0.␣1.
C␣␣0.000000␣␣0.000000␣␣0.000000
End␣of␣basis
␣&SCF
Title
␣Carbon␣atom,␣start␣orbitals
ITERATIONS
␣20,␣20
Occupied
␣2␣0␣0␣0␣0␣0␣0␣0
␣&RASSCF
Title
␣Carbon␣atom.
Symmetry
␣4
Spin
␣3
nActEl
␣2␣0␣0
Frozen
␣0␣0␣0␣0␣0␣0␣0␣0
Inactive
␣2␣0␣0␣0␣0␣0␣0␣0
Ras2
␣0␣1␣1␣0␣0␣0␣0␣0
LevShft
␣0.00
LumOrb
```

```
Thrs
␣0.1d-8␣0.1d-4␣0.1d-4
␣&MOTRA
Title
␣Carbon␣atom.
LumOrb
Frozen
␣␣␣␣1␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0
␣&GUGA
Title
␣Carbon␣atom.
Electrons
␣␣␣␣4
Spin
␣␣␣␣3
Symmetry
␣␣␣␣8
Inactive
␣␣␣␣1␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0
Active
␣␣␣␣0␣␣␣␣1␣␣␣␣1␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0␣␣␣␣0
CiAll
␣␣␣␣4
␣&MRCI
Title
␣Carbon␣atom
SDCI
>>LINK␣FORCE␣$Project.RunFile␣RUN001
>>LINK␣FORCE␣$Project.RunFile␣RUN002
>>LINK␣FORCE␣$Project.OneInt␣␣ONE001
>>LINK␣FORCE␣$Project.OneInt␣␣ONE002
>>LINK␣FORCE␣$Project.RasOrb␣␣NAT001
>>LINK␣FORCE␣$Project.CiOrb␣␣␣NAT002
␣&GENANO
Title
␣Carbon␣atom
Project
sets
␣2
Center
C
Weights
␣0.5␣0.5
>>RM␣ONE001␣ONE002␣␣NAT001␣NAT002
```

8.16 GRID_IT

8.16.1 Description

GRID_IT is an interface program for calculations of molecular orbitals and density in a set of cartesian grid points. The calculated grid can be visualized by molcas gv program.

8.16.2 Dependencies

The GRID_IT program requires the communication file *RUNFILE*, produced by GATEWAY and an orbital file *INPORB*: *SCFORB*, *RASORB*, *PT2ORB*, generated by program SCF(calculated with RHF or UHF hamiltonian), RASSCF, or CASPT2, respectively.

8.16.3 Files

Below is a list of the files that are used/created by the program `GRID_IT`.

Input files

File	*Contents*
RUNFILE	File for communication of auxiliary information generated by the program `GATEWAY`, or `SEWARD`. Contains many things, e.g. the basis set specifications and is augmented with specific entries by most of the individual programs (also by `SCF`).
INPORB	*SCFORB* or another orbitals file (*RASORB*, *CIORB*, *CPFORB*, *SIORB*, *PT2ORB*) containing calculated orbitals. If used after `SCF` run, the information about one-electron energies is also retrieved.

Output files

File	*Contents*
GRID	Output grid file, with default extension grid - the only file needed for drawing program. By default this file is binary. In the case of UHF calculation, `GRID_IT` produces two files: *a.grid* and *b.grid* with grids for alpha and beta electrons. `MOLCAS GV` code could be used to combine grid files for visualization of total or spin density.

8.16.4 Input

Normally, it is no reason to change any default setting of calculated grid - the choice of appropriate grid size, net frequency, as well as choice of MO can be done automatically.

Below follows a description of the input to `GRID_IT`. The keywords are always significant to four characters, but in order to make the input more transparent, it is recommended to use the full keywords. The `GRID_IT` program section of the *MOLCAS* input starts with the standard reference to the code:

```
␣&GRID_IT
```

Argument(s) to a keyword are always supplied on the next line of the input file, except when explicitly stated otherwise.

Optional general keywords

Keyword	*Meaning*
TITLe	One line following this one is regarded as title.
NAME	One line following this one is used for generation of grid filename in the form: 'Project.Name.grid'.
FILE	On the following line user can specify the filename, which will be used instead of INPORB (default).
SPARse	Set up sparse cartesian net with 1 grid point per a.u. Note that quality of the grid can be poor. Default (without Sparse or Dense) is 3 points per a.u.
DENSe	Set up net with 10 grid points per a.u. Note that using this option without choice of orbitals to draw you can produce very large output file.
PACK	Use packing of data, to create a lower quality, but smaller output files.
NOPACK	Do not use packing of data. By default, unless Dense grid is used, the data is packed, so the picture has lower (screen) quality.
GAP	Keyword, followed by real equals to distance between the atomic nuclei in the molecule and the border of grid. Default value is 4.0 a.u.
ORBItal	Direct specification of orbitals to show. Next line set up number of calculated grids. And at next line(s) pairs of integers - symmetry and orbital within this symmetry is given.
SELEct	Direct specification of orbitals to show. Follows by one line in the format: symmetry:first_orbital-last_orbital (Ex: 1:2-7 2:5-8)
NoSort	Do not sort orbitals by occupation numbers and orbital energies
ORANge	Followed by 2 numbers, to limit the interval of orbitals by occupation numbers.
ERANge	Followed by 2 numbers, to limit the interval of orbitals by one-electron energies
ALL	Calculate grids for all molecular orbitals. Using this keyword you can produce a huge output file!
TOTAl	Request to calculate a grid for the (correct) total density computed from contributions of all orbitals, instead of (default) just from the orbitals chosen by user.

VB	This keyword enables plotting of the orbitals from the latest `CASVB` orbitals. Note that the appropriate *`RASORB`* orbitals must be available in the *`INPORB`* file.
ATOM	Calculate density at the position of atoms.
ASCII	Keyword for ASCII format of output file. NOTE that `GV` automatically recognizes both (binary and ASCII) formats. This keyword is useful if a calculation of the grid file and visualization should be done on computers with different architecture. Also note, that if you use native MS Windows compilers to build `GV` you have to pass resulting grid file via unix2dos.
CUBE	Keyword to produce a set of ASCII files for MOLDEN visualization. To import the file to MOLDEN, select 'Read Cube' in density mode.
NPOINTS	Keyword, followed by 3 integers equal to number of grid points in x, y, z directions. Using for non-automatic choice of grid network.
GRID	Keyword to set manually coordinates of a grid. Followed by number of cartesian coordinates, and on next lines - x y z coordinates of a grid (in a.u.)

Input example

An example for high quality picture, containing selected orbitals:

```
␣&GRID_IT
Title
␣Water
Dense
Orbital
2
2␣1
4␣1
```

An example for screen quality picture, containing all orbitals:

```
␣&GRID_IT
SPARSE
PACK
ALL
```

```
␣&GRID_IT
Title
␣Water
*␣to␣draw␣all␣partially␣occupied␣orbitals
ORange
0.01␣1.99
```

8.17 GUESSORB

8.17.1 Description

The GUESSORB program generates a start guess for orbitals. The file *GSSORB* is created containing these orbitals. They are also put to the runfile and their presence is automatically detected by the programs SCF and RASSCF if needed.

8.17.2 Dependencies

The GUESSORB program requires that the one electron file *ONEINT* as well as the communication file *RUNFILE* exist. These are generated by the program SEWARD

8.17.3 Files

Below is a list of the files that are used/created by the program GUESSORB.

Input files

File	*Contents*
ONEINT	One-electron integral file generated by the program SEWARD.
RUNFILE	File for communication of auxiliary information generated by the program SEWARD. Contains many things, e.g. the basis set specifications and is augmented with specific entries by most of the individual programs (also by GUESSORB).

Output files

File	*Contents*
GSSORB	GUESSORB orbital output file. Contains a start guess for orbitals.
RUNFILE	Communication file for subsequent programs.
MOLDEN	Molden input file for molecular orbital analysis.

8.17.4 Input

Below follows a description of the input to `GUESSORB`. The keywords are always significant to four characters, but in order to make the input more transparent, it is recommended to use the full keywords. The input for each module is preceded by its name like:

```
␣&GUESSORB
```

Argument(s) to a keyword are always supplied on the next line of the input file, except explicitly stated otherwise.

Keywords

All keywords are optional except for END OF INPUT which is mandatory.

Keyword	*Meaning*
PRMO	This keword will make `GUESSORB` print the orbitals that are generated. On the next line an integer is to be specified that control how much output you get, see below. On the same line you can optionally specify a floating point number that control how many orbitals are printed. Only orbitals with orbital energy less than this number will be printed, default is 5.0au. 1 — Only occupation numbers and orbital energies are printed. 2 — As for 1 but with an additional sorted list of orbital energies. 3 — As for 2 but with orbitals printed in compact format. 4 — As for 3 but orbitals are printed in full format.
PRPOPULATION	This keyword will print a mulliken population analysis based on the assumptions guessorb make with regards to populating orbitals. No further input is necessary.
STHR	This keyword controls how many orbitals will be deleted. On the next line you specify a threshold that have the default 1×10^{-5}. The overlap matrix is diagonalized and only eigenvectors with eigenvalues larger that this threshold will be used, the other will be deleted. This removes near liner dependence.
TTHR	This keyword controls how many orbitals will be deleted. On the next line you specify a threshold that have the default 1×10^{6}. The kinetic energy matrix is diagonalized in the space of virtual orbitals and only orbitals with energies below this threshold is used, the other will be deleted. This removes degrees of freedom describing core correlation.
GAPTHR	This keyword controls how guessorb attempt to populate the orbitals. On the next line a threshold is specified that have the default 0.01. Using this threshold guessorb will make a closed shell configuration

if it can find a HOMO/LUMO gap that is larger than the specified threshold. If that can not be done, guessorb will place a number of orbitals in an active space in such a way that the gap between the three spaces (inactive, active and secondary) will be larger than the threshold.

END OF INPUT

Input examples

In this example `GUESSORB` is used to produce a mulliken population based on assumptions that are made for population of orbitals.

```
␣&GUESSORB
PrPopulation
End␣of␣Input
```

In this example `GUESSORB` is used to construct an active space for `RASSCF` if there are near degeneracies around the fermi level. The orbital file that is produced can be fed directly into `RASSCF` without specifying the active space.

```
␣&GUESSORB
GapThr
␣0.5
End␣of␣Input
␣&RASSCF
LumOrb
End␣of␣input
```

8.18 GUGA

The `GUGA` program generates coupling coefficients used in the `MRCI` and the `CPF` programs in Direct CI calculations[97]. These coupling coefficients are evaluated by the Graphical Unitary Group Approach[98]–[99], for wavefunctions with at most two electrons excited from a set of reference configurations. The program was written by P. E. M. Siegbahn, Institute of Physics, Stockholm University, Sweden. Only the `MRCI` program can use several reference configurations. The reference configurations can be specified as a list, where the occupation numbers are given for each active orbital (see below) in each reference configuration, or as a Full CI within the space defined by the active orbitals. In the `GUGA`, `MRCI` and `CPF` programs, the orbitals are classified as follows: Frozen, Inactive, Active, Secondary, and Deleted orbitals. Within each symmetry type, they follow this order. For the `GUGA` program, only the inactive and active orbitals are relevant.

- **Inactive:** Inactive orbitals are doubly occupied in all reference configurations, but excitations out of this orbital space are allowed in the final CI wavefunction, i.e., they are correlated but have two electrons in all *reference* configurations. Since only single and double excitations are allowed, there can be no more than two holes in the active orbitals. Using keyword NoCorr (See input description) a subset of the inactive orbitals can be selected, and at most a single hole is then allowed in the selected set. This allows the core-polarization part of core-valence correlation, while preventing large but usually inaccurate double-excitation core correlation.

- **Active:** Active orbitals are those which may have different occupation in different reference configurations. Using keyword OneOcc (See input description) a restriction may be imposed on some selection of active orbitals, so that the selected orbitals are always singly occupied. This may be useful for transition metal compounds or for deep inner holes.

8.18.1 Dependencies

The `GUGA` program does not depend on any other program for its execution.

8.18.2 Files

Input files

The `GUGA` program does not need any input files apart from the file of input keywords.

Output files

File	*Contents*
CIGUGA	This file contains the coupling coefficients that are needed in subsequent CI calculations. For information about how these coefficients are structured you are referred to the source code[99]. The theoretical background for the coefficient can be found in Refs [98]–[99] and references therein.

Intermediate files

File	*Contents*
TEMP01	This is a temporary file that is needed while generating the coupling coefficients. It is of no use to other programs and is automatically discarded by MOLCAS.

8.18.3 Input

This section describes the input to the `GUGA` program in the MOLCAS program system, with the program name:

```
␣&GUGA
```

The first four characters of the keywords are decoded and the rest are ignored.

Keywords.

Formally, there are no compulsory keyword. Obviously, some input must be given for a meaningful calculation.

Keyword	*Meaning*
TITLe	The lines following this keyword are treated as title lines, until another keyword is encountered.
SPIN	The spin degeneracy number, i.e. 2S+1. The value is read from the line following the keyword, in free format. The default value is 1, meaning a singlet wave function.
ELECtrons	The number of electrons to be correlated in the CI of CPF calculation. The value is read from the line following the keyword, in free format. Note that this number should include the nr of electrons in inactive orbitals. An alternative input specification is NACTEL. Default: Twice nr of inactive orbitals.
NACTel	The number of electrons in active orbitals in the reference configurations. The value is read from the line following the keyword, in free format. Note that this number includes only the of electrons in active orbitals. An alternative input specification is ELECTRONS. Default: Zero.
INACtive	The number of inactive orbitals, i.e. orbitals that have occupation numbers of 2 in all reference configurations. Specified for each of the symmetries. The values are read from the line following the keyword, in free format.
ACTIve	The number of active orbitals, i.e. orbitals that have varying occupation numbers in the reference configurations. Specified for each of the symmetries. The values are read from the line following the keyword, in free format. At least one of the Inactive or Active keywords must be present for a meaningful calculation. If one of them is left out, the default is 0 in all symmetries.
ONEOcc	Specify a number of active orbitals per symmetry that are required to have occupation number one in all configurations. These orbitals are the first active orbitals. The input is read from the line after the keyword, in free format.
NOCOrr	Specify the number of inactive orbitals per symmetry out of which at most one electron (total) is excited. These orbitals are the first inactive orbitals. The input is read from the line after the keyword, in free format.
REFErence	Specify selected reference configurations. The additional input that is required usually spans more than one line. The first line after

	the keyword contains the number of reference configurations, and the total number of active orbitals, and these two numbers are read by free format. Thereafter the input has one line per reference configuration, specifying the occupation number for each of the active orbitals, read by 80I1 format. Note that Reference and CIall are mutually exclusive.
CIALL	Use a Full CI within the subspace of the active orbitals as reference configurations. The symmetry of the wavefunction must be specified. The value is read from the line following the keyword, in free format. Note that CIall and Reference are mutually exclusive. One of these two alternatives must be chosen for a meaningful calculation.
FIRST	Perform a first order calculation, i.e. only single excitations from the reference space. No additional input is required.
NONINTERACTING SPACE	By default, those double excitations from inactive to virtual orbitals are excluded, where the inactive and virtual electrons would couple to a resulting triplet. With the NonInteracting Space option, such 'non-interacting' configurations are included as well.
PRINT	Printlevel of the program. Default printlevel (0) produces very little output. Printlevel 5 gives some information that may be of interest. The value is read from the line following the keyword, in free format.

Input example

```
␣&GUGA
Title
␣Water␣molecule.
␣2OH␣correlated.
Electrons
␣␣␣␣4
Spin
␣␣␣␣1
Active
␣␣␣␣2␣␣␣␣2␣␣␣␣0␣␣␣␣0
Interacting␣space
Reference
␣␣␣␣3␣␣␣␣4
2020
0220
2002
```

8.19 LOCALISATION

8.19.1 Description

The `LOCALISATION` program of the molcas program system generates localized occupied orbitals according to one of the following procedures: Pipek-Mezey[107], Boys[108, 109], Edmiston-Ruedenberg[110], or Cholesky[111]. Orthonormal, linearly independent, local orbitals may also be generated from projected atomic orbitals (Cholesky PAOs)[111].

Orbital localisation makes use of the fact that a Hartree-Fock wave function is invariant under unitary transformations of the occupied orbitals,

$$\tilde{C}_{\mu i} = \sum_j C_{\mu j} U_{ji}, \tag{8.6}$$

where $\mathbf{U}$ is unitary (i.e. orthogonal for real orbitals). The same is true for the inactive or active orbitals in a CASSCF wave function. Whereas the Pipek-Mezey[107], Boys[108, 109], and Edmiston-Ruedenberg[110] procedures define $\mathbf{U}$ through an iterative maximisation of a localisation functional, the Cholesky orbitals are simply defined through the Cholesky decomposition of the one-electron density, i.e.

$$\sum_i \tilde{C}_{\mu i} \tilde{C}_{\nu i} = P_{\mu\nu} = \sum_i C_{\mu i} C_{\mu i}. \tag{8.7}$$

Cholesky orbitals are thus not optimum localised orbitals by any of the Pipek-Mezey, Boys, or Edmiston-Ruedenberg measures, but rather inherit locality from the density matrix, see [111] for details.

Although these localisation schemes are mostly meant for localising occupied orbitals (except for PAOs which are defined for the virtual orbitals), the `LOCALISATION` program will attempt to localise any set of orbitals that the user specifies. This means that it is possible to mix occupied and virtual orbitals and thereby break the Hartree-Fock invariance. The default settings, however, do not break the invariance.

For Pipek-Mezey, Boys, and Edmiston-Ruedenberg localisations, iterative optimizations are carried out. We use the η-steps of Subotnik *et al.*[112] for Edmiston-Ruedenberg, whereas the traditional Jacobi sweeps (consecutive two-by-two orbital rotations)[107, 112] are employed for the Pipek-Mezey and Boys schemes.

8.19.2 Dependencies

The `LOCALISATION` program requires the one-electron integral file *`ONEINT`* and the communications file *`RUNFILE`*, which contains, among other data, the basis set specifications processed by `GATEWAY` and `SEWARD`. In addition, the Edmiston-Ruedenberg procedure requires the presence of Cholesky decomposed two-electron integrals produced by `SEWARD`.

8.19.3 Files

Below is a list of the files that are used/created by the program `LOCALISATION`.

Input files

File	*Contents*

ONEINT	One-electron integral file generated by the program `SEWARD`.
RUNFILE	File for communication of auxiliary information generated by the program `SEWARD`. Contains many things, e.g. the basis set specifications and is augmented with specific entries by most of the individual programs.
INPORB	File containing the orbitals which are to be localised.
*CHVEC**	Files containing Cholesky vectors (separate files for each irreducible representation, the base file name being CHVEC1 for irreducible representation 1, CHVEC2 for irreducible representation 2, and so forth. Each file CHVEC1, CHVEC2, etc. may be a multifile, hence the asterisk in the base name above). Only needed for Edmiston-Ruedenberg localisation.
*CHRED**	File for storing index arrays for addressing Cholesky vectors. Only needed for Edmiston-Ruedenberg localisation.
*CHORST**	File containing information about the Cholesky decomposition procedure. Only needed for Edmiston-Ruedenberg localisation.

Output files

File	*Contents*
LOCORB	Localized orthonormal orbital output file. Note that LOCORB contains all orbitals (localized as well as non-localized according to the input specification).
DPAORB	Linearly dependent nonorthonormal projected atomic orbital output file (only produced for PAO runs).
IPAORB	Linearly independent nonorthonormal projected atomic orbital output file (only produced for PAO runs).
RUNFILE	Communication file for subsequent programs.
MOLDEN	Molden input file for molecular orbital analysis.

8.19.4 Input

Below follows a description of the input to `LOCALISATION`. The keywords are always significant to four characters, but in order to make the input more transparent, it is recommended to use the full keywords. The `LOCALISATION` program section of the *MOLCAS* input is bracketed by a preceding program reference

```
␣&LOCALISATION
```

Arguments to a keyword are always supplied on the next line of the input file, except explicitly stated otherwise.

Optional general keywords

Keyword	*Meaning*
NORBITALS	The following line specifies the number of orbitals to localise in each irreducible representation. The default is to localise all occupied orbitals as specified in the INPORB input file, except for PAO runs where all the virtual orbitals are treated by default.
NFROZEN	The following line specifies the number of orbitals to freeze in each irreducible representation. The default is not to freeze any orbitals, except for the localisations of the virtual space (see keywords PAO and VIRTual) where the default is to freeze all occupied orbitals (occupation number different from zero, as reported in the INPORB file).
FREEZE	Implicit frozen core option. The default is not to freeze any orbitals, except for the localisations of the virtual space (see keywords PAO and VIRTual) where the default is to freeze all occupied orbitals (occupation number different from zero, as reported in the INPORB file). The definition of core orbitals is taken from program `SEWARD`.
OCCUPIED	Requests that the occupied orbitals should be localised. This is the default except for PAO where the default is virtual.
VIRTUAL	Requests that the virtual orbitals should be localised. The default is to localise the occupied orbitals, except for PAO where the default is virtual.
PIPEK-MEZEY	Requests Pipek-Mezey localisation. This is the default.
BOYS	Requests Boys localisation. The default is Pipek-Mezey.
EDMISTON-RUEDENBERG	Requests Edmiston-Ruedenberg localisation. The default is Pipek-Mezey. Note that this option requires that the Cholesky (or RI/DF) representation of the two-electron integrals has been produced by `SEWARD`.
CHOLESKY	Requests Cholesky localisation (non-iterative). The default is Pipek-Mezey. This and PAO are the only options that can handle point group symmetry. The decomposition threshold is by default 1.0D-8 but may be changed through the THREshold keyword.
PAO	Requests PAO localisation (non-iterative) using Cholesky decomposition to remove linear dependence. The default is Pipek-Mezey. This and Cholesky are the only options that can handle point group symmetry. The decomposition threshold is by default 1.0D-8 but may be changed through the THREshold keyword.
SKIP	Leaves the input orbitals unchanged. It is turned off by default.

ITERations	The following line specifies the maximum number of iterations to be used by the iterative localisation procedures. The default is 100.
THREshold	The following line specifies the convergence threshold used for changes in the localisation functional. The default is 1.0D-6.
THRGradient	The following line specifies the convergence threshold used for the gradient of the localisation functional. The default is 1.0D-2.
THRRotations	The following line specifies the screening threshold used in the Jacobi sweep optimization algorithm. The default is 1.0D-10.
CHOStart	Requests that iterative localisation procedures use Cholesky orbitals as initial orbitals. The default is to use the orbitals from INPORB directly.
ORDEr	Requests that the localised orbitals are ordered in the same way as the Cholesky orbitals would be. This is mainly useful when comparing orbitals from different localisation schemes. The ordering is done according to maximum overlap with the Cholesky orbitals. The default is not to order.
DOMAin	Requests orbital domains and pair domains are set up and analysed. The default is not to set up domains.
THRDomain	The following line specifies two thresholds to be used in defining orbital domains. The first is the Mulliken population threshold such that atoms are included in the domain until the population (divided by 2) is larger than this number (default: 9.0d-1). The second threshold is used for the Pulay completeness check of the domain (default: 2.0d-2).
THRPairdomain	The following line specifies three thresholds to be used for classifying pair domains: R1, R2, and R3. (Defaults: 1.0d-10, 1.0d1, and 1.5d1.) If R is the smallest distance between two atoms in the pair domain (union of the individual orbital domains), then pair domains are classified according to: $R \leq R1$: strong pair, $R1 < R \leq R2$: weak pair, $R2 < R \leq R3$: distant pair, and $R3 < R$: very distant pair.
LOCNatural orbitals	This keyword is used to select atoms for defining the localized natural orbitals (LNOs), thus a set of localized orbitals with well-defined occupation numbers. All other options specified in the `LOCALISATION` program input apply (e.g., input orbitals, localization method, etc.). On the next line give the number of atoms that identify the region of interest and the threshold used to select the localized orbitals belonging to this region (recommended values ¿ 0.2 and ¡ 1). An additional line gives the names of the (symmetry unique) atoms as defined in the Seward input. The keyword LOCN is used to define suitable occupation numbers for RASSCF active orbitals that have been localized. It has proven useful in Effective Bond Order (EBO) analysis. Here is a sample input for a complex containing an iron-iron multiple bond.

```
␣LOCN
␣2␣␣0.3
␣Fe1␣␣Fe2
```

In this example, the (localized) orbitals constructed by the `LOCALISATION` program are subdivided in two groups: those having less than 0.3 total Mulliken population on the two iron atoms, and the remaining orbitals, obviously localized on the iron-iron region. The resulting density matrices for the two subsets of orbitals are then diagonalized separately and the corresponding (localized) natural orbitals written to LOCORB with the proper occupation numbers. Note that the two sets of LNOs are mutually non-orthogonal.

LOCCanonical orbitals This keyword is used to select atoms for defining the localized canonical orbitals (LCOs), thus a set of localized orbitals with well-defined orbital energies (eigenvalues of a local Fock matrix). Please, refer to the analogous keyword LOCN in this manual for more details and input examples.

Limitations

The limitations on the number of basis functions are the same as specified for `SEWARD`.

Input examples

This input is an example of the Boys localisation of the CO molecule. Note that no symmetry should not be used in any calculation of localised orbitals except for Cholesky and PAO orbitals.

```
␣&SEWARD
Basis␣Set
O.STO-3G....
O␣␣0.0␣0.0␣0.0
End␣of␣Basis
Basis␣Set
C.STO-3G....
C␣␣0.0␣0.0␣2.132
End␣of␣Basis
␣&SCF
Title
CO,␣STO-3g␣Basis␣set
Occupied
7
␣&LOCALISATION
Boys
```

This input is an example of the Projected Atomic Orbital localisation of the virtual orbitals of the CO molecule. The threshold for the Cholesky decomposition that removes linear dependence is set to 1.0d-14.

```
␣&SEWARD
Basis␣Set
O.STO-3G....
```

```
O␣␣0.0␣0.0␣0.0
End␣of␣Basis
Basis␣Set
C.STO-3G....
C␣␣0.0␣0.0␣2.132
End␣of␣Basis
␣&SCF
Title
CO,␣STO-3g␣Basis␣set
Occupied
7
␣&LOCALISATION
PAO
Threshold
1.0d-14
```

This input is an example of the Cholesky localisation (using default 1.0d-12 as threshold for the decomposition) of the valence occupied orbitals of the CO molecule. Orbital domains are set up and analyzed.

```
␣&SEWARD
Basis␣Set
O.STO-3G....
O␣␣0.0␣0.0␣0.0
End␣of␣Basis
Basis␣Set
C.STO-3G....
C␣␣0.0␣0.0␣2.132
End␣of␣Basis
␣&SCF
Title
CO,␣STO-3g␣Basis␣set
Occupied
7
␣&LOCALISATION
Cholesky
Freeze
Domain
```

8.20 LOPROP

The program `LOPROP` is a new tool to compute molecular properties based on the one-electron density or transition-density and one-electron integrals like charges, dipole moments and polarizabilities. `LOPROP` allows to partition such properties into atomic and interatomic contributions. The method requires a subdivision of the atomic orbitals into occupied and virtual basis functions for each atom in the molecular system. It is a requirement for the approach to have any physical significance that the basis functions which are classified as "occupied" essentially are the atomic orbitals of each species. It is therefore advisable to use an ANO type basis set, or at least a basis set with general contraction.

The localization procedure is organized into a series of orthogonalizations of the original basis set, which will have as a final result a localized orthonormal basis set. **Note that this module does not operate with symmetry.**

A static property, which can be evaluated as an expectation value, like a charge, a component of the dipole moment or an exchange-hole dipole moment, is localized by transforming the integrals of the property and the one-electron density matrix to the new basis and restricting the trace to the subspace of functions of a single center or the combination of two centers.

The molecular polarizability is the first order derivative of the dipole moment with respect to an electric field and the localized molecular polarizability can be expressed in terms of local responses. In practical terms a calculation of localized polarizabilities will require to run seven energy calculations. The first one is in the absence of the field and the other six calculations are in the presence of the field in the ± x,y,z axes respectively.

For a detailed description of the method and its implementation see [113].

8.20.1 Dependencies

The dependencies of the `LOPROP` module is the union of the dependencies of the `SEWARD`, and the program used to perform the energy calculation, namely the `SCF`, `RASSCF`, or `CASPT2` module. The user can also provide `LOPROP` with a density matrix as input; then `LOPROP` only depends on `SEWARD`. The one-electron transition density matrix can also be localized to compute, for example, Förster transition probabilities; then `LOPROP` depends on `RASSI` to compute the transition density.

8.20.2 Files

The files of the `LOPROP` module is the union of the files of the `SEWARD` module, and the `SCF` or `RASSCF`, or `CASPT2` module. An exception is if a density matrix is given as input or when a transition density matrix is localized, see below.

Input files

File	*Contents*
USERDEN	The density matrix given as input when the keyword USERDENSITY is included in the input. The density matrix should be of the following form: triangularly stored ((1,1),(2,1),(2,2),(3,1) etc.) with all off-diagonal elements multiplied by two.
USERDEN1	The density matrix for a field-perturbed calculation (X = +delta)
USERDEN2	The density matrix for a field-perturbed calculation (X = -delta)
USERDEN3	The density matrix for a field-perturbed calculation (Y = +delta)
USERDEN4	The density matrix for a field-perturbed calculation (Y = -delta)
USERDEN5	The density matrix for a field-perturbed calculation (Z = +delta)
USERDEN6	The density matrix for a field-perturbed calculation (Z = -delta)
TOFILE	The one-electron transition density matrix, which optionally can be put to disk by `RASSI`, see its manual pages.

Output files

In addition to the standard output unit `LOPROP` will generate the following file.

File	*Contents*
MpProp	File with the input for `NEMO`.

8.20.3 Input

This section describes the input to the `LOPROP` program. The program name is:

```
␣&LOPROP
```

Keywords

There are no compulsory keywords.

Keyword	*Meaning*
NOFIeld	The calculation is run in the absence of a field and only static properties like charges and dipole moments are computed. The default is to go beyond the static properties.
DELTa	The magnitude of the electric field in the finite field perturbation calculations to determine the polarizabilities. Default value is 0.001 au.
ALPHa	A parameter in the penalty function used for determining the charge fluctuation contribution to the polarizabilities. See eq. 17 in [113]. The default value of 7.14 is good for small molecules (less than 50 atoms). For larger molecules, a smaller alpha (e.g. 2.0) may be needed for numerical stability.
BOND	Defines the maximum allowed bond length based on the ratio compared to Bragg-Slater radii. All contributions in bonds longer than this radius will be redistributed to the two atoms involved in the bond, so the the total molecular properties are left unaltered. The default value is 1.5.
MPPRop	Defines the maximum l value for the multipole moments written to the MpProp file. If the value specified is larger than the highest multipole moment calculated it will be reset to this value, which is also the default value. The 'MULTipoles' keyword in Seward can change the default value.

EXPAnsion center Defines which points will be used as the expansion centers for the bonds. The next line must contain either 'MIDPoint' in order just to use the midpoint of the bond or 'OPTImized' in order to let LoProp move the expansion center along the bond. The latter is still highly experimental!

USERdensity No density matrix is computed instead it is read as an input from the file *USERDEN*. This enables LOPROP to obtain localized properties for densities that currently can not be computed with *MOLCAS*. If the keyword NOFIeld is not given, six additional files are required (*USERDEN1*-*USERDEN6*), each containing the density matrix of a perturbed calculation, see above. Observe the form of *USERDEN*, see above.

TDENsity This keyword signals that the one-electron density matrix which is to be read comes from the TOFILE file generated by RASSI. The keyword is followed by two integers that gives number of initial and final state of the transition. For example, if it is the transition density between the first and second state which should be localized, the integers should be 1 and 2. The keyword implies NOFIeld

XHOLe The exchange hole dipole moment is computed, localized and given as additional output. This quantity can be used to compute local dispersion coefficients according to Becke and Johnson.[114] The numerical integration routine in *MOLCAS*is used.

Input example

Below follows an example input to determine the localized charges, and dipole moments of acetone at the CASSCF level of theory.

```
␣&GATEWAY
Title
acetone
Basis␣set
O.ANO-L...3s2p1d.
O1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.2975761001
End␣of␣basis
Basis␣set
H.ANO-L...2s1p.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.2337732815␣␣␣␣␣␣␣-3.5130421867
H2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-2.2337732815␣␣␣␣␣␣␣-3.5130421867
H3␣␣␣␣␣␣␣␣␣␣␣␣␣␣-1.6626924186␣␣␣␣␣␣␣␣3.5885858729␣␣␣␣␣␣␣-0.9131174874
H4␣␣␣␣␣␣␣␣␣␣␣␣␣␣-1.6626924186␣␣␣␣␣␣␣-3.5885858729␣␣␣␣␣␣␣-0.9131174874
H5␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣1.6626924186␣␣␣␣␣␣␣␣3.5885858729␣␣␣␣␣␣␣-0.9131174874
H6␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣1.6626924186␣␣␣␣␣␣␣-3.5885858729␣␣␣␣␣␣␣-0.9131174874
End␣of␣basis
Basis␣set
C.ANO-L...3s2p1d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣2.4832019942␣␣␣␣␣␣␣-1.4632998706
C2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣-2.4832019942␣␣␣␣␣␣␣-1.4632998706
C3␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣␣␣0.0000000000
End␣of␣basis
␣&SEWARD
␣&SCF
ITERATIONS
```

```
20
Occupation
15
␣&RASSCF
SPIN
1
SYMMETRY
1
NACTEL
4␣0␣0
INACTIVE
13
RAS2
4
␣&LOPROP
NoField
Expansion␣Center
Optimized
Bond
1.5
MpProp
2
```

In case the density matrix is given as input the input is of the form below (where $OldDir is a variable defined by the user pointing to the directory where the input density is).

```
␣&Gateway
Basis␣Set
O.6-31G*..10s4p1d.3s2p1d.
o␣-0.00434␣0.0␣0.00039␣/Angstrom
End␣of␣Basis␣Set
Basis␣Set
H.6-31G*..4s.2s.
h2␣-0.764837␣0.0␣0.588254␣/Angstrom
h3␣0.761705␣0.0␣␣0.580812␣/Angstrom
End␣of␣Basis␣Set
␣&Seward
>>COPY␣$OldDir/Density␣$WorkDir/$Project.UserDen
␣&LoProp
UserDensity
```

8.21 `M2SO`

The `M2SO` is used to read and transform information produced by the `RASSCF` and `MOTRA` modules into a format that can be used by the spin-orbit CI program `EPCISO` developped by the Toulouse and Lille groups. This program can be obtained by sending an e-mail to:
Valérie Vallet
Université des Sciences et Technologies de Lille – CNRS
Laboratoire PhLAM
UFR de Physique
F-59655 Villeneuve d'Ascq
E-mail: valerie.vallet@univ-lille1.fr
Using the MOLPRO keyword you can also output data for the `MOLPRO` quantum chemistry package.

8.21.1 Running `M2SO` for `EPCISO`

To run the `EPCISO`, you have to run `MOLCAS` program as follows:

1. Run `SEWARD` to generate one and two-electron integrals over the symmetry adapted orbitals (SAOs). The generated ***RUNFILE*** contains information on the number of atoms, number of shells, number of atomic orbitals and the basis sets.
2. Run the program `RASSCF` to generate the file ***RASORB*** which contains the molecular orbitals on the basis of SAOs.
3. Run `MOTRA` to transforms one and two-electron integrals from SAOs to MO basis.
4. Run the interface `M2SO`. It generates the following files:
   - ***prefix.geom***: file containing information on the atoms, *i.e.* atomic coordinates, nuclear charge, the basis sets.
   - ***prefix.oms***: file containing the molecular orbitals.
   - ***prefix.f25***: file containing the symmetry of each molecular orbital and the crossing table of the abelian group.
   - ***prefix.ijkl***: file containing the one and two-electron integrals (coulombic and exchange parts).
   - ***prefix.amfi***: file containing the mean-field SO integrals for the AMFI program.

   The `M2SO` program reorders the MOs so that all occupied orbitals appear first (definition of the Fermi sea). The new ordering of the MOs is printed in the output file. Here is how the input looks like:

   ```
    &M2SO
   End Of Input
   ```

8.21.2 Running `M2SO` for `MOTRA`

If you run the `M2SO` program before `MOTRA`, proceed as follows

1. Run `MOTRA` with the keyword `INT;` and `GBASIS;`
2. Copy the `MOTRA` output file to ***$Project.molpro.out*** into the work directory
3. Run `M2SO` with the MOLPRO keyword and it generates the following files:

- ***prefix.hcore***: contains the one-electron Hamiltonian on the AO basis
- ***prefix.molporb***: contains the molecular orbitals to be used as guess in `MOLPRO`.

Here is how the input looks like:

```
 &M2SO
MOLPRO
End Of Input
```

8.21.3 Dependencies

To run the program `M2SO` successfully the one- and two-electron integrals transformed to the MO basis by `MOTRA` are needed. In addition, a set of MO coefficients must be available.

8.21.4 Files

Input files

The following is a list of input files

File	*Contents*
ONEINT	One-electron integral file generated by `SEWARD`.
TRAONE	Transformed one-electron integrals.
*TRAINT**	Transformed two-electron integrals.
INPORB	MO's read in formatted form.

Input orbitals must be supplied in the form of a formatted ASCII file *INPORB*.

8.21.5 Output file

File	*Contents*
GEOMFIL	file containing information on the atoms, *i.e.* atomic coordinates, nuclear charge, the basis sets.
OMSFIL	file containing the molecular orbitals.
SYMFIL	file containing the symmetry of each molecular orbital and the crossing table of the abelian group.
MONFIL	file containing the one-electron Hamiltonian
IJKFIL	file containing the reordered one and two-electron integrals (coulombic and exchange parts).
TRAMFI	file containing the mean-field SO integrals generated by the `AMFI` program within `SEWARD`.
MPROHO	file containing the one-electron Hamiltonian on the AO basis for `MOLPRO`.
MPROORB	file containing the molecular orbitals for `MOLPRO`.

8.21.6 Input

The input for each module is preceded by its name like:

```
 &M2SO
```

Keywords

Following the list of keywords of which the first four characters are recognized while the rest is ignored.

Keyword	*Meaning*
MOLPRO	This keyword is used to produce the files *`MPROHO`* and *`MPROORB`* for the `MOLPRO` program. By default, `M2SO` writes only files for the `EPCISO` program.
END OF INPUT	Marks the end of the input.

Input example

The following input will produce files for MOLPRO:

```
 &M2SO
MOLPRO
End of input
```

8.22 MBPT2

8.22.1 Description

The `MBPT2` program of the *MOLCAS* program system computes the second order correlation energy and the reference weight for a closed-shell Hartree-Fock reference wave function, based on a Møller-Plesset partitioning of the Hamiltonian and canonical orbitals.

If `SEWARD` performed a Cholesky decomposition of the two-electron integrals prior to running the `MBPT2` program, Cholesky vectors will be employed for computing the second order energy correction. This is done by first transforming the Cholesky vectors to MO basis and subsequently computing the $(ai|bj)$ integrals. These integrals are either computed, stored on disk, and then read back in to memory during the energy evaluation (i.e. mimicking a conventional calculation) or they may be computed on-the-fly. The user may choose either algorithm through the Cholesky-specific options described below.

If `SEWARD` did not perform a Cholesky decomposition, the transformation of the two-electron integrals in AO basis ($\mathcal{O}(N^4)$, where N is the number of basis functions) to the exchange operator matrices $\mathbf{K}^{ij}$ in MO basis ($\mathcal{O}(O^2)$ matrices of size V^2, where O and V denote the number of occupied and virtual orbitals, respectively), is either done conventionally, using the two-electron integral file *`ORDINT`*, which was generated in a previous step by the `SEWARD` integral code.

8.22.2 Dependencies

The `MBPT2` program requires the communications file *RUNFILE*. It contains specifications processed by `SEWARD`, the Hartree-Fock canonical orbitals, eigenvalues and energy generated by `SCF`. For Cholesky-based calculations, all Cholesky related files (see the manual pages for `SEWARD`) must be available, whereas for conventional (not integral-direct) calculations the two-electron integral file *ORDINT* is required. Hence, before running `MBPT2`, a `SEWARD` and a `SCF` run have to be performed.

8.22.3 Files

Input files

File	*Contents*
RUNFILE	File for communication of auxiliary information generated by the program `SEWARD` and updated by the program `SCF`. Contains many things, e.g. the basis set specifications processed by `SEWARD` and the Hartree-Fock orbitals/eigenvalues/energy computed by `SCF`, and is augmented with specific entries by most of the individual programs (also by `MBPT2`).
*ORDINT**	Ordered and packed two-electron integrals generated by the program `SEWARD` (eventually segmented into multiple files). Only used for conventional (not integral-direct) runs.
*CHVEC**	Files for storing Cholesky vectors. Only used for Cholesky runs.
*CHRED**	File for storing index arrays for addressing Cholesky vectors. Only used for Cholesky runs.
*CHORST**	File containing information about Cholesky decomposition procedure Only used for Cholesky runs.
CHOR2F	File containing the mapping between Cholesky vector storage and the canonical orbital ordering. Only used for Cholesky runs.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
*MOLINT**	Resulting file of transformed integrals. Scratch file; conventional calculation only.

*LUHLFn**	n=1 to 3. Intermediate files used in the 1st, 2nd, and 3rd, respectively, transformation step. Conventional calculation only.

Output files

File	*Contents*
RUNFILE	File for communication of auxiliary information.

8.22.4 Input

Below follows a description of the input to `MBPT2`. The keywords are always significant to four characters, but in order to make the input more transparent, it is recommended to use the full keywords. The input for each module is preceded by its name like:

```
␣&MBPT2
```

Argument(s) to a keyword are always supplied on the next line of the input file, except explicitly stated otherwise. No compulsory keywords are required for `MBPT2`. The reference statement mentioned above is sufficient for a default `MBPT2` run.

Optional keywords

Keyword	*Meaning*
TITLE	All lines following this line are regarded as title lines until the next keyword is encountered. A maximum of ten lines is allowed.
PRINT	Specifies the general print level of the calculation. An integer has to be supplied as argument. The default value, 0, is recommended for production calculations.
FREEZE	Specifies the total number of frozen occupied orbitals. The lowest-energy occupied orbitals are then automatically identified and frozen. The keyword takes as argument one integer. Incompatible with the FROZEN keyword.
FROZEN	Specifies the number of frozen occupied orbitals in each of the irreducible representations (irreps) of the subgroup of D_{2h} in which the system is represented. The counting of the orbitals follows the *increasing* orbital energy within each irrep, with those orbitals being frozen first that correspond to lowest orbital energies. The keyword takes as argument *nIrrep* (# of irreps) integers. Incompatible with the FREEZE keyword. Default is to freeze non-valence orbitals.

DELETED	Specifies the number of deleted orbitals in each of the irreducible representations (irreps) of the subgroup of D_{2h} in which the system is represented. The counting of the orbitals follows the *decreasing* orbital energy within each irrep, with those orbitals being deleted first that correspond to highest orbital energies. The keyword takes as argument *nIrrep* (# of irreps) integers. **OBS.:** Those orbitals, which have been deleted already in the `SCF` calculation (cf. SPDELETE, OVLDELETE of the `SCF` program description) are never seen by the `MBPT2` program and hence are **not** to be deleted again with the present option.
SFROZEN	Allows to specify specific orbitals to freeze in each of the irreducible representations (irreps) of the subgroup of D_{2h} in which the system is represented. In the 1st line after the keyword the number of orbitals to freeze for each irrep is specified (*nIrrep* (# of irreps) integers). The next $\leq$ *nIrrep* lines reference the orbitals to freeze for the related irrep, following an enumeration of the individual orbitals of 1, 2, 3,... according to *increasing* orbital energy. Note that the orbital reference numbers obey the original ordering and also include those orbitals which may have been frozen already by the FROZEN or FREEZE options. If the corresponding irrep does not contain any specific orbitals to freeze (i.e. a zero was supplied for this irrep in the 1st line), no line orbital reference input line is supplied for that irrep.
SDELETED	Allows to specify specific orbitals to delete in each of the irreducible representations (irreps) of the subgroup of D_{2h} in which the system is represented. In the 1st line after the keyword the number of orbitals to delete for each irrep is specified (*nIrrep* (# of irreps) integers). The next $\leq$ *nIrrep* lines reference the orbitals to delete for the related irrep, following an enumeration of the individual orbitals of 1, 2, 3,... according to *increasing* orbital energy. Note that the orbital reference numbers obey the original ordering. If the corresponding irrep does not contain any specific orbitals to freeze (i.e. a zero was supplied for this irrep in the 1st line), no line orbital reference input line is supplied for that irrep.
LUMORB	Molecular orbital coefficients and energies read from INPORB file rather than RunFile.
EREF	Specifies the value of the reference energy. Available only in combination with LumOrb. Default value of the reference energy is set to zero.
TEST	If this keyword is specified the input is checked without performing any calculation.
LOVMP2	"Freeze-and-Delete" type of MP2, available only in connection with Cholesky or RI. An example of input for the keyword LOVM is the following:

```
LovMP2
```

```
2␣␣0.2␣␣(nCenters,thrs)
C1␣N␣␣␣␣(Center␣labels)
DoMP2
```

In this case, both occupied and virtual orbitals (localized by the program) are divided in two groups: those (A) mainly located on the two (symmetry independent) centers C1 and C2, and the remaining ones (B), which are obviously “outside” this region. The value of the threshold (between 0 and 1) is used to perform this selection (in the example, 20% of the gross Mulliken population of a given orbital on the specified atoms). By default, the MP2 calculation is performed only for the correlating orbitals associated with the region A (“active site”). The keyword DoMP2 is optional and forces the program to perform also an independent MP2 calculation on the “frozen region” (B). Alternatively, one can specify the keyword VirAll in order to use all virtual orbitals as correlating space for the occupied orbitals of the active site.

Optional keywords specific to Cholesky calculations

Observe that these keywords are disregarded if the integrals were not Cholesky decomposed by `SEWARD`. Furthermore, they are disregarded for algorithm 0 (see below).

Keyword	*Meaning*
CHOAlgorithm	Takes as argument one positive integer specifying the algorithm to use for Cholesky MP2. Options: 0 [generate MO integrals on disk from Cholesky vectors], 1 [compute integrals on-the-fly, minimal operation count, level 2 BLAS], 2 [compute integrals on-the-fly, not minimal operation count, level 3 BLAS], Default is 2.
VERBose	Increases printing from the Cholesky MP2 routines, although not by much. Default is (almost) no printing.
DECOmpose	Requests Cholesky decomposition of the $(ai\|bj)$ integrals. Unless user-defined (see below), the threshold used is identical to that used by `SEWARD` for decomposing the two-electron integrals. Default is to not decompose.
THRCholesky	Specifies the threshold for $(ai\|bj)$ Cholesky decomposition. Takes as argument one real number. Default is the threshold used by `SEWARD` for decomposing the two-electron integrals.
NODEcompose	Turns off Cholesky decomposition of the (ai—bj) integrals. Default is to not decompose.
SPAN	Specifies the span factor used for $(ai\|bj)$ Cholesky decomposition. Takes as argument one real number. Default is the span factor used by `SEWARD` for decomposing the two-electron integrals.

MXQUal	Specifies the max. number of qualified diagonals treated during $(ai\|bj)$ Cholesky decomposition. Takes as argument one integer. Default is 10% of the max. rank of $(ai\|bj)$, although never more than 200.
PRESort	Presort the MO Cholesky vectors according to the batches over occupied orbitals. This will reduce the amount of I/O performed during on-the-fly assembly of the $(ai\|bj)$ integrals. This keyword is obsolete.

Limitations

The maximum number of selectively frozen SFRO or selectively deleted orbitals SDEL in each symmetry is limited to 50.
The limitations on the number of basis functions are the same as specified for `SEWARD`.

Input example

```
␣&MBPT2
Title
␣H2O:␣␣O(9.5/4.2),␣H(4/2)␣R=Re
*␣The␣lowest␣energy␣occupied␣orbital␣in␣the␣repr.␣no.1␣will␣be␣frozen␣in
*␣MBPT2␣calculations.␣The␣number␣of␣representations␣is␣4␣and␣all␣zeros
*␣must␣be␣explicitly␣given
Frozen
1␣0␣0␣0
*␣Two␣highest␣energy␣external␣orbitals␣in␣the␣repr.␣no.3␣will␣be␣deleted
*␣in␣MBPT2␣calculations.␣The␣number␣of␣representations␣is␣4␣and␣all
*␣zeros␣must␣be␣explicitly␣given
Deleted
0␣0␣2␣0
*␣One␣occupied␣orbital␣in␣symmetry␣no.1␣will␣be␣additionally␣frozen␣by
*␣using␣the␣SFRO␣option.␣Let␣it␣be␣the␣third␣SCF␣occupied␣orbital␣in
*␣this␣symmetry
sFrozen
1␣0␣0␣0␣␣␣␣Gives␣the␣number␣of␣frozen␣orbitals␣in␣each␣symmetry
3␣␣␣␣␣␣␣␣␣␣Gives␣the␣frozen␣orbital␣reference␣number␣in␣symmetry␣no.␣1
*␣Two␣external␣orbitals␣in␣symmetry␣no.1␣and␣one␣external␣orbital␣in
*␣symmetry␣3␣will␣be␣deleted.␣In␣symmetry␣1␣let␣it␣be␣the␣second␣and
*␣third␣external␣orbitals,␣and␣in␣symmetry␣3␣the␣third␣(already␣deleted
*␣in␣by␣using␣the␣option␣DELE)␣external␣orbital
sDeleted
2␣0␣1␣0␣␣␣␣Gives␣the␣number␣of␣orbitals␣to␣be␣deleted␣in␣each␣symmetry
2␣3␣␣␣␣␣␣␣␣Gives␣the␣reference␣numbers␣of␣external␣orbitals␣in␣sym.␣1
3␣␣␣␣␣␣␣␣␣␣Gives␣the␣reference␣number␣of␣the␣external␣orb.␣in␣sym.␣3
END␣Of␣Input
```

8.23 MCKINLEY

8.23.1 Description

This program calculates the second and first order derivatives used for calculating second order derivatives of the energies with perturbation dependent basis sets for SCF and CASSCF wavefunctions. Vibronic frequencies are calculated by the `MCLR` program which `MCKINLEY` will start automatically.

`MCKINLEY` is written such that properties can be computed for any kind of valence basis function that `SEWARD` will accept. Note, that `MCKINLEY` can't handle ECP type basis sets.

`MCKINLEY` is able to compute the following integral derivatives:

- overlap integrals,
- kinetic energy integrals,
- nuclear attraction integrals,
- electron repulsion integrals,
- external electric field integrals,

`MCKINLEY` employs two different integration schemes to generate the one- and two-electron integral derivatives. The nuclear attraction and electron repulsion integrals are evaluated by a modified Rys-Gauss quadrature [115]. All other integral derivatives are evaluated with the Hermite-Gauss quadrature. The same restriction of the valence basis sets applies as to `SEWARD`. None of the integral 2nd derivatives are written to disk but rather combined immediately with the corresponding matrix from the wavefunction. Integral 1st derivative information is preprocessed and written to disc for later use in `MCLR`.

Note that `MCKINLEY` for frequency calculations normally automatically starts the `MCLR` with a default input. To override this check the keyword NOMCLR!

At present the following limitations are built into `MCKINLEY`:

Max number of unique basis functions:	2000
Max number of symmetry independent centers:	500
Highest angular momentum:	15
Highest symmetry point group:	D_{2h}

8.23.2 Dependencies

`MCKINLEY` depends on the density, Fock matrices and Molecular orbitals generated by `SCF` or `RASSCF`. In addition it needs the basis set specification defined in `SEWARD`. These dependencies, however, are totally transparent to the user.

8.23.3 Files

Input files

Apart from the standard input unit `MCKINLEY` will use the following input files.

File	*Contents*
RUNFILE	Auxiliary data
RYSRW	Data base for the fast direct evaluation of roots and weights of the Rys polynomials. This file is a part of the program system and should not be manipulated by the user.
ABDATA	Data base for the evaluation of roots and weights of high order Rys polynomial. This file is a part of the program system and should not be manipulated by the user.

Output files

In addition to the standard output unit `MCKINLEY` will generate the following files.

File	*Contents*
MCKINT	File with information needed by the `MCLR` response program.

8.23.4 Input

Below follows a description of the input to `MCKINLEY`.

In addition to the keywords and the comment lines the input may contain blank lines. The input for each module is preceded by its name like:

```
␣&MCKINLEY
```

The first four characters of the keywords are decoded while the rest are ignored. However, for a more transparent input we recommend the user to use the full keywords.

Compulsory keywords

Keyword	*Meaning*
END Of Input	This marks the end of the input to the program.

Optional keywords

Keyword	*Meaning*
ONEOnly	Compile only the nuclear repulsion and one-electron integrals contribution. The default is to compute all contributions.
CUTOff	Threshold for ignoring contributions follows on the next line. The default is 1.0D-7. The prescreening is based on the 2nd order density matrix and the radial overlap contribution to the integral derivatives.

SHOWHESSIAN	The Hessian contributions will be printed.
PERTURBATION	This key word takes one of the arguments HESSian or GEOMetry on the following line. Hessian makes `MCKINLEY` compute the full geometrical Hessian, which is required for a subsequent calculation of the vibronic frequencies with the `MCLR` program. GEOMetry calculates only those parts of the geometrical Hessian that correspond to symmetry allowed displacements (Could be used in a second order geometry optimization). The default is to compute the Hessian for vibrational frequency calculations.
NOMCLR	Request that this calculation is not followed by the execution of the MCLR module.
VERBOSE	The output will be a bit more verbose.

The following is an example of an input which will work for almost all practical cases.

```
␣&MCKINLEY
Perturbation
Hessian
End␣of␣input
```

8.24 MCLR

The `MCLR` program in *MOLCAS* performs response calculations on single and multiconfigurational SCF wave functions with the technique described in [115]. The right hand side (RHS) and thus the perturbation has to be defined through a preceding `MCKINLEY` calculation. Second order derivatives are obtained from a `MCKINLEY` and a consecutive `MCLR` calculation, with a geometrical displacement as the external perturbation. If the response of a geometrical perturbation is calculated, harmonic frequencies corresponding to the most abundant masses are printed. `MCLR` also calculates isotope shifted frequencies. Per default, vibrational frequencies are calculated for all possible single isotopic substitutions. `MCLR` always calculates the response of an electric field and prints the polarizability.

Note that the user should not normally, for frequency calculations, request the execution of this module since this will be automatic.

The `MCLR` code also calculates the Lagrangian multipliers required for a SA-MCSCF single state gradient [116], where the RHS is generated by the program it self. Through an `MCLR` and a consecutive `ALASKA` calculation, analytical gradients of a SA-MCSCF state may be obtained. **Note that `ALASKA` will automatically run the `MCLR` module!** Thus, with `STRUCTURE` geometry optimizations of excited MCSCF states can be performed (See input example below).

The `MCLR` program is based on the split GUGA formalism. However, it uses determinant based algorithms to solve the configuration interaction problem [117], in analogy to how it is done in the `RASSCF`. For spin symmetric wave function ($M_S = 0$) the time reversal symmetry is used, and the innermost loops are performed in combinations instead of determinant.

The upper limit to the size of the CI wave function that can be handled with the present program is about the same as for the `RASSCF`. The present version of the code is just able to handle CASSCF wave function, RASSCF wave function will soon be included.

The orbital handling is based on a one index transformation technique. The integrals is the transformed to occupied orbitals in two indexes, this can be done directly or from disk based integrals generated by `SEWARD`.

8.24.1 Dependencies

To start the `MCLR` module the one-electron integrals generated by `SEWARD` have to be available. Moreover, `MCLR` requires the wave function from a `SCF` or `RASSCF` calculation and apart from in an evaluation of SA-MCSCF gradients, it also requires the differentiated integrals from `MCKINLEY`.

8.24.2 Files

Input files

File	*Contents*
ONEINT	Contains the one-electron integrals
*ORDINT**	Contains the two-electron integrals.
RUNFILE	This file carries all information needed to start up the integral evaluation section in direct calculations.
ABDATA	Data base for the evaluation of roots and weights of high order Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
RYSRW	Data base for the fast direct evaluation of roots and weights of the Rys polynomials. This file is a part of the program system and should not be manipulated by the user.
JOBIPH	The binary input file which has been generated by the `RASSCF` program This file carries the results of the wave function optimization such as MO- and CI-coefficients.

Output files

File	*Contents*
MCKINT	Communication file between `MCLR` and `MCKINLEY` and `RASSI`

UNSYM	ASCII file where all essential information, like geometry, Hessian normal modes and dipole derivatives are stored.
MLDNFQ	Molden input file for harmonic frequency analysis.

Scratch files

File	*Contents*
TEMP0x	x=1,8 used for for integral transformation and storing half transformed integrals.
REORD	Used for storing data used in the transformation of CI vectors from determinant base to CSF base.
TEMPCIV	Exchange file for temporary storing the CI vectors during the PCG.
RESP	Binary bile where the solution of the response equations are stored.
JOPR	Used for half transformed integrals in direct mode.
KOPR	Used for half transformed integrals in direct mode.

8.24.3 Input

This section describes the input to the `MCLR` program in the *MOLCAS* program system. The input for each module is preceded by its name like:

```
␣&MCLR
```

Compulsory keywords

The following input has to be included in all calculations.

Keyword	*Meaning*
END OF INPUT	This marks the end of the input data.

Optional keywords

A list of these keywords is given below:

Keyword	*Meaning*
SALA	Makes MCLR compute the Lagrangian multipliers for a state average MCSCF wave function. These multipliers are required by `ALASKA` to obtain analytical gradients for an excited state, when the excited state is determined by a SA optimization. SALA has to be followed by an integer on the next line, specifying the excited state for which the gradient is required. This integer specifies which root in the SA room the multipliers are calculated for.
EXPDIMENSION	Here follows the dimension of the explicit Hamiltonian used as preconditioner in the Preconditioned conjugate gradient algorithm. Default 100.
ITERATIONS	Specify the maximum number of iterations in the PCG. Default 50.
LOWMEMORY	Lowers the amount of memory used, by paging out the CI vectors on disk. This will lower the performance, but the program will need less memory.
PRINT	Raise the print level, default 0.
RASSI	This keyword is used for transforming the CI vectors to split GUGA representation, and transforming the orbital rotations to AO basis, to make the response accessible for state interaction calculations.
SEWARD	Specify one particle operators, used as right hand side, form the ONEINT file constructed by `SEWARD` The keyword is followed by one row for each perturbation: LABEL symmetry Component
ENDSEWARD	Marks the end of perturbation specifications read from seward ONEINT file.
THRESHOLD	Specify the convergence threshold for the PCG. Default is 1.0e-04.
DISOTOPE	Calculates frequencies modified for double isotopic substitution.
THERMOCHEMISTRY	Request an user specified thermochemical analysis. The keyword must be followed by a line containing the Rotational Symmetry Number, a line containing the Pressure (in atm), and lines containing the Temperatures (in K) for which the thermochemistry will be calculated. The section is ended by the keyword "End of PT".
TIME	Calculates the time dependent response of an electric periodic perturbation. The frequency of the perturbation should be specified on the following line. Used to calculated time dependent polarizabilities and required in a `RASSI` calculation of two photon transition moments.

MASS	Used to generate single and double (in conjunction with DISO) isotope shifted frequencies, with the isotope masses specified by the user. This implementation can be useful for example in calculating intermolecular frequencies which are contaminated by the BSSE. By setting the corresponding masses to the very large numbers, ghost orbitals can be used in the frequency calculation. MASS needs the atomic label and the new mass in units of u (real), for each element of the molecule.

Input example

A default input for a harmonic frequency calculation.

```
␣&MCLR
End␣of␣input
An␣input␣for␣a␣harmonic␣frequency␣calculation␣␣with␣modified␣isotopic␣masses
for␣hydrogen␣and␣oxygen.


␣&MCLR
MASS
H
2.0079
O
150000.998
End␣of␣input
```

Thermochemistry for an asymmetric top (Rotational Symmetry Number = 1), at 1.0 atm and 273.15, 298.15, 398.15 and 498.15 K.

```
␣&MCLR
THERmochemistry
␣1
␣1.0
␣273.15
␣298.15
␣398.15
␣498.15
End␣of␣PT
End␣of␣input
```

The time dependent response is calculated for a perturbation of frequency 0.2 au.

```
␣&MCLR
TIME
0.2
End␣of␣input
```

The time dependent response is calculated for a perturbation of frequency 0.2 au.

```
␣&MCLR
SALA
2
End␣of␣input
```

Computes the Lagrangian multipliers for state number 2 in the SA room. Note, that 2 refers to the SA room. Thus, if the ground state is not included in the SA, the numbering of roots in the ci room and SA room differ. With the following `RASSCF` input

```
␣&RASSCF
CiRoot
␣2␣3
␣2␣3
␣1␣1
RlxRoot
2
End␣of␣input
```

SALA 2 yields the gradient for ci root number 3. Geometry optimization of an excited SA-CASSCF state can be done normally using `AUTO`, and requires the use of the RLXR keyword in the `RASSCF` input to specify the selected root to be optimized. An explicit input to `MCLR` is not required but can be specified if default options are not appropriate.

8.25 `MOTRA`

The program `MOTRA` is used to transform one- and two-electron integrals from AO to MO basis. It reads the one-electron file and the file of ordered and symmetry blocked two-electron integrals generated by `SEWARD`.

The two-electron integral transformation is performed one symmetry block at a time, as a series of four sequential one-index transformations. The process includes a sorting of the half transformed integrals prior to the second half transformation. This step is performed in core if there is space enough of memory available to keep one symmetry block of integrals. Otherwise the half transformed integrals are written out on an temporary file. The result of the transformation is two files, *`TRAONE`* and *`TRAINT`* which contain the transformed one- and two- electron integrals, respectively.

The one-electron transformation is performed for the kinetic integrals and the bare nuclei Hamiltonian. If there are frozen orbitals `MOTRA` replaces the bare nuclei Hamiltonian by an effective Fock operator, which incorporates the interaction between the frozen (core) electrons and the remaining electrons. In practice this means that in any subsequent calculation (for example `MRCI`, `CPF` or `MBPT`) the effect of the frozen orbitals is incorporated into the one-electron Hamiltonian, and these orbitals need not be explicitly accounted for. The total energy of the frozen electrons is added to the nuclear-nuclear repulsion energy and transferred from `MOTRA` to the subsequent program(s).

The two-electron transformation is performed from the list of ordered integrals generated by `SEWARD` (file *`ORDINT`*).

8.25.1 Dependencies

To run the program `MOTRA` successfully the one- and two-electron integrals are needed. In addition, a set of MO coefficients must be available. The latter may be obtained by any wave function optimization program.

8.25.2 Files

Input files

The following is a list of input files

File	*Contents*
ONEINT	One-electron integral file generated by SEWARD.
*ORDINT**	Ordered two-electron integrals generated by SEWARD.
INPORB	If MO's are read in formatted form.
JOBIPH	If molecular orbitals are read from a RASSCF interface.

In general, input orbitals are supplied in the form of a formatted ASCII file, but can also be taken directly from the binary interface file, *JOBIPH*, created by the RASSCF program. The selection in controlled by input options.

Output files

The program MOTRA creates two files: The first carries all basic information and a list of transformed one-electron integrals. The second file includes the transformed two-electron integrals.

The following is a list of output files

File	*Contents*
TRAONE	Auxiliary data and transformed one-electron integrals.
*TRAINT**	Transformed two-electron integrals.

Intermediate files

MOTRA generates one intermediate file with half transformed one-electron integrals, *LUHALF*. It is scratched at the end of the run. This file can be large in calculations with extended basis sets. It is used to store one symmetry block of integrals at a time.

The following is a list of local files

File	*Contents*
*LUHALF**	Auxiliary data and transformed one-electron integrals.

8.25.3 Input

This section describes the input to the MOTRA program in the *MOLCAS* program system. The input for each module is preceded by its name like:

```
␣&MOTRA
```

The first four characters of the keywords are decoded while the rest are ignored.

Compulsory keywords

The following keywords are compulsory.

Keyword	*Meaning*
LUMOrb	Specifies that the molecular orbitals are read from a formatted file produced by one of the wave function generating programs. **Note** that either of Lumorb or Jobiph should be specified. LUMORB is the default keyword. No additional input is required.
JOBIph	Specifies that the molecular orbitals are read from a `RASSCF` job interface file. `MOTRA` will in this case read the average orbitals. No additional input is required.
END of input	Marks the end of the input.

When natural orbitals from a RASSCF (or a state averaged CASSCF) calculation are to be used in `MOTRA`, they can be produced, or extracted from an existing *`JOBIPH`* file, by `RASSCF`, using keyword OUTOrbitals.

Optional keywords

There are a few useful optional keywords that can be specified. The following is a list

Keyword	*Meaning*
AUTO	This keyword specified automatic deletion of orbitals based on occupation numbers. The following line contain one threshold per symmetry, and all orbitals with occupation numbers smaller that the threshold will be deleted. If AUTO and DELEte are both specified, the larger number will be used.
DELEted	Specifies the number of virtual orbitals that are not to be used as correlating orbitals in the subsequent CI calculation. The last orbitals in each symmetry are deleted. The default is no deleted orbitals. One additional line with the number of deleted orbitals in each symmetry (free format).
FROZen	Specifies the number of doubly occupied orbitals that are left uncorrelated in subsequent correlation calculation(s). Additional orbitals can be frozen in these programs, but from an efficiency point of view it is preferable to freeze orbitals in the transformation. One additional line with the number of frozen orbitals in each symmetry (free format). For more details on freezing orbitals in `MOTRA` see the program description. The frozen orbitals are the first in each symmetry block. Default is to freeze the core (but not semi-core) orbitals.
ONEL	Specifies that only one-electron integrals are to be transformed. No additional input is required.

PRINt	Specifies the print level in the program. The default (1) does not print the orbitals that are used in the transformation, but they appear at print level 2. Beware of large print levels since vast amounts of output may be produced. The value is read from the line after the keyword, in free format.
RFPErt	This keyword will add a constant reaction field perturbation to the bare nuclei Hamiltonian. The perturbation is read from the file ***RUNFILE*** and is the latest self consistent perturbation generated by one of the programs `SCF` or `RASSCF`.
TITLe	This keyword should be followed by exactly one title line.

Input example

```
␣&MOTRA
Title
␣Water␣molecule
*␣Don’t␣correlate␣1s␣on␣oxygen
Frozen
␣1␣0␣0␣0
Lumorb
End␣of␣input
```

8.26 MRCI

The `MRCI` program generates Multi Reference SDCI or ACPF[96] wavefunctions. ACPF is a modification of the CPF[94] method which allows more than one reference configuration. The program is based on the Direct CI method[97], and with the coupling coefficients generated with the Graphical Unitary Group Approach[98]–[99]. (See program description for `GUGA`). If requested, `MRCI` computes matrix elements of those one-electron properties for which it can find integrals in the ***ONEINT*** file. It also generates natural orbitals that can be fed into the property program to evaluate certain one electron properties. The natural orbitals are also useful for Iterated Natural Orbital (INO) calculations.

The `MRCI` code is a modification of an MRCI program written by M. Blomberg and P. E. M. Siegbahn (Institute of Physics, Stockholm University, Sweden), which has later been extensively modified (P.-Å. Malmqvist)

The program can calculate several eigenvectors simultaneously, not necessarily those with lowest eigenvalue. However, in the ACPF case, only one single eigenvector is possible.

Orbital subspaces

The orbital space is divided into the following subspaces: Frozen, Inactive, Active, Secondary, and Deleted orbitals. Within each symmetry type, they follow this order.

- **Frozen:** Frozen orbitals are always doubly occupied, i.e., they are not correlated. Orbitals may be frozen already in the integral integral transformation step, program

MOTRA, but can also be specified in the input to the MRCI program. The former method is more efficient, and has the effect that the frozen orbitals are effectively removed from the subsequent MRCI calculation.

- **Inactive:** Inactive orbitals are doubly occupied in all reference configurations, but excitations out of this orbital space are allowed in the final CI wavefunction, i.e., they are correlated but have two electrons in all reference configurations. Restrictions may be applied to excitation from some inactive orbitals, see keyword NoCorr in the GUGA input section.
- **Active:** Active orbitals are those which may have different occupation in different reference configurations. Restrictions may be applied to occupation of some active orbitals, see keyword OneOcc in the GUGA input section.
- **Secondary:** This subspace is empty in all reference configurations, but may be populated with up to two electrons in the excited configurations. This subspace is not explicitly specified, but consists of the orbitals which are left over when other spaces are accounted for.
- **Deleted:** This orbital subspace does not participate in the CI wavefunction at all. Typically the 3s,4p,... components of 3d,4f..., or orbitals that essentially describe core correlation, are deleted. Similar to freezing, deleting can be done in MOTRA, which is more efficient, but also as input specifications to the MRCI program.

Since ordinarily the frozen and deleted orbitals were handled by MOTRA and the subdivision into inactive and active orbitals were defined in GUGA, the only time one has to specify orbital spaces in the input to MRCI is when additional frozen or deleted orbitals are required without recomputing the transformed integrals.

8.26.1 Dependencies

The program needs the coupling coefficients generated by the program GUGA and transformed one- and two-electron integrals generated by the program MOTRA.

8.26.2 Files

Input files

File	*Contents*
CIGUGA	Coupling coefficients from GUGA.
*TRAINT**	Transformed two-electron integrals from MOTRA.
TRAONE	Transformed one-electron integrals from MOTRA.
ONEINT	One-electron property integrals from SEWARD.
MRCIVECT	Used for input only in restart case.

Output files

File	*Contents*
CIORBnn	One or more sets of natural orbitals, one for each CI root, where nn stands for 01,02, etc.
MRCIVECT	CI vector, for later restart.

Note that these file names are the FORTRAN file names used by the program, so they have to be mapped to the actual file names. This is usually done automatically in the *MOLCAS* system. However, in the case of several different numbered files *CIORBnn* only the first will be defined as default, with the FORTRAN file name *CIORB* used for *CIORB01* .

Local files

File	*Contents*
FTxxF001	MRCI produces a few scratch files that are not needed by any other program in *MOLCAS*. Presently, these are xx=14, 15, 16, 21, 23, 25, 26, 27, and 30. The files are opened, used, closed and removed automatically. See source code for further information.

8.26.3 Input

This section describes the input to the `MRCI` program in the *MOLCAS* program system, with the program name:

```
␣&MRCI
```

Keywords

The first four characters are decoded and the rest are ignored.

Keyword	*Meaning*
TITLE	The lines following this keyword are treated as title lines, until another keyword is encountered. A maximum of ten lines is allowed.
SDCI	This keyword is used to perform an ordinary Multi-Reference Singles and Doubles CI, MR-SDCI, calculation. This is the default assumption of the program. Note that SDCI and ACPF are mutually exclusive. No additional input is required.

ACPF	This keyword tells the program to use the Average Coupled Pair Functional, ACPF, when computing the energy and natural orbitals. Note that SDCI and ACPF are mutually exclusive. No additional input is required.
GVALue	The coefficient g which is used in the ACPF functional. The default value is = 2.0/(Nr of correlated electrons). The user-supplied value is entered on the next line.
NRROots	Specifies the number of CI roots (states) to be simultaneously optimized. The default is 1. The value is read from the next line.
ROOTs	Specifies which root(s) to converge to. These are defined as the ordinal number of that eigenvector of the reference CI which is used as start approximation. The default is the sequence 1,2,3… The values are entered on the next line(s). If the number of roots is larger than 1, it must first have been entered using keyword NRROOTS. The keywords ROOTS and SELECT are mutually exclusive.
SELEct	Another way of specifying the roots: instead of using ordinal numbers, the roots selected will be those NRROOTS which have largest projections in a selection space which is specified on the next lines, as follows: One line gives NSEL, the number of vectors used to define the selection space. For each selection vector, there is on the following line(s) a specification read by the *FORTRAN* statement READ(*,*) NC,(CSEL(JJ+J),SSEL(JJ+J),J=1,NC) where NC is the number of CSF-s, and CSEL and SSEL are pairs of text strings and coefficients. The text string is composed of the digits 0,1,2,3 and denotes the GUGA case numbers of the active orbitals, defining uniquely a CSF belonging to the reference space. The keywords ROOTS and SELECT are mutually exclusive.
RESTart	Restart the calculation from a previous calculation. No additional input is required. The ***MRCIVECT*** file is required for restarted calculations.
THRPrint	Threshold for printout of the wavefunction. All configurations with a coefficient greater than this threshold are printed. The default is 0.05. The value is read from the line following the keyword.
ECONvergence	Energy convergence threshold. The result is converged when the energy of all roots has been lowered less than this threshold in the last iteration. The default is 1.0d-8. The value is read from the line following the keyword.
PRINt	Print level of the program. Default is 5. The value is read from the line following the keyword.
MAXIterations	Maximum number of iterations. Default 20. The value is read from the line following the keyword. The maximum possible value is 49.

MXVECTORS	Maximum number of trial vector pairs (CI+sigma) kept on disk. Default is MAX(NRROOTS,10). It should never be smaller than NRROOTS. A good value is 3*NRROOTS or more.
TRANSITION	This keyword is relevant to a multi-root calculation. In addition to properties, also the transition matrix elements of various operators, for each pair of wave functions, will be computed.
FROZEN	Specify the number of orbitals to be frozen in **addition** to the orbitals frozen in the integral transformation. The values are read from the line following the keyword. Default is 0 in all symmetries.
DELETED	Specify the number of orbitals to be deleted in **addition** to the orbitals deleted in the integral transformation. The values are read from the line following the keyword. Default is 0 in all symmetries.
REFCI	Perform only reference CI. No additional input is required.
PRORBITALS	Threshold for printing natural orbitals. Only orbitals with occupation number larger than this threshold appears in the printed output. The value is read from the line following the keyword. Default is 1.0d-5.

Input example

```
␣&MRCI
Title
␣Water␣molecule.
␣1S␣frozen␣in␣transformation.
Sdci
```

8.27 MULA

The MULA calculates intensities of vibrational transitions between electronic states.

8.27.1 Dependencies

The MULA program may need one or more UNSYM files produced by the MCLR program, depending on input options.

8.27.2 Files

Input files

File	*Contents*
UNSYM	Output file from the MCLR program

Output files

File	*Contents*
`plot.intensity`	Contains data for plotting an artificial spectrum.

8.27.3 Input

The input for `MULA` begins after the prgoram name:

```
␣&MULA
```

There are no compulsory keyword. The optional keywords can be abbreviated down to four characters. An unidentified keyword makes the program stop. Most keywords are followed by additional input data. When a keyword is followed by mandatory lines of input, this sequence cannot be interrupted by a comment line. NOTE: If you have already been using earlier versions of this program, you will find that some keywords have changed.

Keywords

Keyword	*Meaning*
TITLE	Followed by a single line, the title of the calculation.
FORCE	A force field will be given as input (or read from file), defining two oscillators for which individual vibrational levels and transition data will be computed.
ATOMS	Followed by one line for each individual atom in the molecule. On each line is the label of the atom, consisting of an element symbol followed by a number. After the label, separated by one or more blanks, one can optionally give a mass number; else, a standard mass taken from the file data/atomic.data. After these lines is one single line with the keyword ”END of atoms”.
INTERNAL	Specification of which internal coordinates that are to be used in the calculation. Each subsequent line has the form ’BOND *a b*’ or ’ANGLE *a b c*’ or or ’TORSION *a b c d*’ or or ’OUTOFPL *a b c d*’, for bond distances, valence angles, torsions (e.g. dihedral angles), and out-of-plane angles. Here, *a*. . . *d* stand for atom labels. After these lines follows one line with the keyword ”END of internal”.
MODES	Selection of modes to be used in the intensity calculation. This is followed by a list of numbers, enumerating the vibrational modes to use. The modes are numbered sequentially in order of vibrational frequency. After this list follows one line with the keyword ”END of modes”.

MXLEvels	Followed by one line with the maximum number of excitations in each of the two states.
VARIational	If this keyword is included, a variational calculation will be made, instead of using the default double harmonic approximation.
TRANsitions	Indicates the excitations to be printed in the output. Followed by the word FIRST on one line, then a list of numbers which are the number of phonons – the excitation level – to be distributed among the modes, defining the vibrational states of the first potential function (force field). Then similarly, after a line with the word SECOND, a list of excitation levels for the second state.
ENERgies	The electronic T_0 energies of the two states, each value is followed by either "eV" or "au".
GEOMetry	Geometry input. Followed by keywords FILE, CARTESIAN, or INTERNAL. If FILE, the geometry input is taken from UNSYM1 and UNSYM2. If CARTESIAN or INTERNAL, two sections follow, one headed by a line with the word FIRST, the other with the word SECOND. For the CARTESIAN case, the following lines list the atoms and coordinates. On each line is an atom label, and the three coordinates (x, y, z). For the INTERNAL case, each line defines an internal coordinate in the same way as for keyword INTERNAL, and the value.
MXORder	Maximum order of transition dipole expansion. Next line is 0, if the transition dipole is constant, 1 if it is a linear function, etc.
OSCStr	If this keyword is included, the oscillator strength, instead of the intensity, of the transitions will calculated.
BROAdplot	Gives the peaks in the spectrum plot an artificial halfwidth. The default lifetime is $130 \cdot 10^{-15}$ s but this can be changed with keyword LIFEtime followd by the value.
NANOmeters	If this keyword is included, the plot file will be in nanometers. Default is in eV.
CM-1	If this keyword is included, the plot file will be in cm^{-1}. Default is in eV.
PLOT	Enter the limits (in eV, cm^{-1}, or in nm) for the plot file.
VIBWrite	If this keyword is included, the vibrational levels of the two states will be printed in the output.
VIBPlot	Two files, plot.modes1 and plot.modes2, will be generated, with pictures of the normal vibrational modes of the two electronic states.
HUGElog	This keyword will give a much more detailed output file.
SCALe	Scales the Hessians, by multiplying with the scale factors following this keyword.

DIPOles	Transition dipole data. If MXORDER=0 (see above), there follows a single line with x, y, z components of the transition dipole moment. If MXORDER=1 there are an additional line for each cartesian coordinate of each atom, with the derivative of the transition dipole moment w.r.t. that nuclear coordinate.
NONLinear	Specifies non-linear variable substitutions to be used in the definition of potential surfaces.
POLYnomial	Gives the different terms to be included in the fit of the polynomial to the energy data.
DATA	Potential energy surface data.

Input example

```
␣&MULA
Title
␣Water␣molecule
Atoms
␣O1
␣H2
␣H3
End␣Atoms
Internal␣Coordinates
␣Bond␣␣O1␣H2
␣Bond␣␣O1␣H3
␣Angle␣H3␣O1␣H2
End␣Internal␣Coordinates
MxLevels
␣␣0␣␣3
Energies
␣First
␣␣0.0␣eV
␣Second
␣␣3.78␣eV
Geometry
␣Cartesian
␣First
␣␣O1␣␣␣␣␣0.0000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣-0.5000000000
␣␣H2␣␣␣␣␣1.6000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣1.1000000000
␣␣H3␣␣␣␣-1.6000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣1.1000000000
␣End
␣Second
␣␣O1␣␣␣␣␣0.0000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣-0.4500000000
␣␣H2␣␣␣␣␣1.7000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣1.0000000000
␣␣H3␣␣␣␣-1.7000000000␣␣␣␣␣␣0.0000000000␣␣␣␣␣␣1.0000000000
␣End
ForceField
␣First␣state
␣Internal
␣␣0.55␣0.07␣0.01
␣␣0.07␣0.55␣0.01
␣␣0.01␣0.01␣0.35
␣Second␣state
␣Internal
␣␣0.50␣0.03␣0.01
␣␣0.03␣0.50␣0.01
␣␣0.01␣0.01␣0.25
DIPOles
```

```
␣␣0.20␣0.20␣1.20
BroadPlot
LifeTime
␣10.0E-15
NANO
PlotWindow
␣260␣305
End␣of␣input
```

```
␣&MULA
TITLe
␣Benzene
ATOMs
␣␣C1
␣␣C2
␣␣C3
␣␣C4
␣␣C5
␣␣C6
␣␣H1
␣␣H2
␣␣H3
␣␣H4
␣␣H5
␣␣H6
End␣of␣Atoms
GEOMetry
␣file
INTERNAL␣COORDINATES
␣Bond␣␣␣␣C1␣C3
␣Bond␣␣␣␣C3␣C5
␣Bond␣␣␣␣C5␣C2
␣Bond␣␣␣␣C2␣C6
␣Bond␣␣␣␣C6␣C4
␣Bond␣␣␣␣C1␣H1
␣Bond␣␣␣␣C2␣H2
␣Bond␣␣␣␣C3␣H3
␣Bond␣␣␣␣C4␣H4
␣Bond␣␣␣␣C5␣H5
␣Bond␣␣␣␣C6␣H6
␣Angle␣␣␣C1␣C3␣C5
␣Angle␣␣␣C3␣C5␣C2
␣Angle␣␣␣C5␣C2␣C6
␣Angle␣␣␣C2␣C6␣C4
␣Angle␣␣␣H1␣C1␣C4
␣Angle␣␣␣H2␣C2␣C5
␣Angle␣␣␣H3␣C3␣C1
␣Angle␣␣␣H4␣C4␣C6
␣Angle␣␣␣H5␣C5␣C3
␣Angle␣␣␣H6␣C6␣C2
␣Torsion␣C1␣C3␣C5␣C2
␣Torsion␣C3␣C5␣C2␣C6
␣Torsion␣C5␣C2␣C6␣C4
␣Torsion␣H1␣C1␣C4␣C6
␣Torsion␣H2␣C2␣C5␣C3
␣Torsion␣H3␣C3␣C1␣C4
␣Torsion␣H4␣C4␣C6␣C2
␣Torsion␣H5␣C5␣C3␣C1
␣Torsion␣H6␣C6␣C2␣C5
END␣INTERNAL␣COORDINATES
VIBPLOT
␣cyclic␣4␣1
ENERGIES
␣First
␣␣0.0␣eV
```

```
␣Second
␣␣4.51␣eV
MODES
␣14␣30␣5␣6␣26␣27␣22␣23␣16␣17␣1␣2␣9␣10
END
MXLE␣-␣MAXIMUM␣LEVEL␣of␣excitation␣(ground␣state␣-␣excited␣state)
␣␣2␣2
MXOR␣-␣MAXIMUM␣ORDER␣in␣transition␣dipole.
␣␣1
OscStr
Transitions
␣First
␣␣0
␣Second
␣␣0␣1␣2
FORCEFIELD
␣First
␣␣␣file
␣Second
␣␣␣file
DIPOLES
␣file
```

8.28 NUMERICAL_GRADIENT

The `NUMERICAL_GRADIENT` module is a program which numerically evaluates the gradient of the energy with respect to nuclear perturbations.

Note that this module is automatically invoked by the `ALASKA` module if the wave function method is MBPT2, CCSDT, CASPT2, MS-CASPT2, or a calculation using the Cholesky decomposition. The user should normally never request the execution of this module; instead it is advised to use the NUMErical keyword in Alaska, if it is necessary to force the use of numerical gradients rather than analytical ones.

The module is parallelized over the displacements, which in case of large jobs gives a linear speed up compared to a serial execution, although in order to obtain this it is important to choose the number of nodes such that the number of contributing perturbations is a multiple of the number of nodes. For a given molecule the number of perturbations equals the number of atoms times 6 (a perturbation with plus and minus delta for each of the three axises). Symmetry can of course reduce this number. If the request of execution originates from the `SLAPAF` module further reduction in perturbations is achieved due to the utilization of rotational and translational invariance.

8.28.1 Dependencies

The dependencies of the `NUMERICAL_GRADIENT` module is the union of the dependencies of the `SEWARD`, `SCF`, `RASSCF`, `MBPT2`, `MOTRA`, `CCSDT`, and `CASPT2` modules.

8.28.2 Files

The files of the `NUMERICAL_GRADIENT` module is the union of the files of the `SEWARD`, `SCF`, `RASSCF`, `MBPT2`, `MOTRA`, `CCSDT`, and `CASPT2` modules.

8.28.3 Input

This section describes the input to the `NUMERICAL_GRADIENT` program. The program name is:

```
␣&Numerical_Gradient
```

Keywords

Keyword	*Meaning*
END OF INPUT	Marks the end of the input to the program.

8.29 PARALLELTEST

The `PARALLELTEST` module simulates a parallel/serial execution of the Molcas modules on user machines.

8.29.1 Description

A algorithm of the paralleltest module does not have not any physical meaning. It tests:

1. reading a input file in parallel/serial mode,
2. synchronization of a file between nodes,
3. parallelization of loops.

For user convenience we provide timing for loops parallelization.

Because, the Molcas is module oriented package, where one parallel module can call another parallel module, the test code executes parallel loops which calls other parallelized subroutines. Then the internal subroutine is executed in serial mode.

From the user point of view it is important to be able to test a loops parallelization according to a size of working vectors. In this simple test program we provide an integer parameter, 'N', to control the number of elements in one dimension of a square matrix. A total number of elements for a square matrix is N*N. The N is optional and should be provided at the input file if needed. By default the N is equal 100. The test module will use a two square real*8 matrices during execution. A results of test are printed to standard output.

8.29.2 Files

Standard input

The `PARALLELTEST` obeys all rules for format of *MOLCAS*'s input. The input is always preceded by the program reference

```
␣&paralleltest
```

Example:

```
&Paralleltest
 ................
```

A input of `PARALLELTEST` module has only one optional and integer parameter, N, to control the # of elements in one dimension of a square matrix.

Example:

```
&PARALLELTEST
 * Define a number of elements, N, in a one dimension of a square matrix. By default N=100;
 200
```

8.30 `RASSCF`

The `RASSCF` program in *MOLCAS* performs multiconfigurational SCF calculations using the Restricted Active Space SCF construction of the wave function [118]. RASSCF is an extension of the Complete Active Space (CAS) approach, in which the wave function is obtained as a full CI expansion in an active orbital space [119, 21]. The RASSCF method is based on a partitioning of the occupied molecular orbitals into the following groups:

- **Inactive orbitals:** Orbitals that are doubly occupied in all configurations.
- **Active orbitals:** These orbitals are subdivided into three separate groups:
    - **RAS1 orbitals:** Orbitals that are doubly occupied except for a maximum number of holes allowed in this orbital subspace.
    - **RAS2 orbitals:** In these orbitals all possible occupations are allowed (former Complete Active Space orbitals).
    - **RAS3 orbitals:** Orbitals that are unoccupied except for a maximum number of electrons allowed in this subspace.

CASSCF calculations can be performed with the program, by allowing orbitals only in the RAS2 space. A single reference SDCI wave function is obtained by allowing a maximum of 2 holes in RAS1 and a maximum of 2 electrons in RAS3, while RAS2 is empty (the extension to SDT- and SDTQ-CI is obvious). Multireference CI wave functions can be constructed by adding orbitals also in RAS2.

The `RASSCF` program is based on the split GUGA formalism. However, it uses determinant based algorithms to solve the configuration interaction problem [117]. To ensure a proper spin function, the transformation to a determinant basis is only performed in the innermost loops of the program to evaluate the σ-vectors in the Davidson procedure and to compute the two-body density matrices. The upper limit to the size of the CASSCF wave function that can be handled with the present program is about 10^7 CSFs and is, in general, limited by the dynamic work array available to the program.

The orbital optimization in the `RASSCF` program is performed using the super-CI method. The reader is referred to the references [118, 120] for more details. A quasi-Newton (QN) update method is used in order to improve convergence. No explicit CI-orbital coupling is used in the present version of the program, except for the coupling introduced in the QN update.

Convergence of the orbital optimization procedure is normally good for CASSCF type wave functions, but problems can occur in calculations on excited states, especially when several states are close in energy. In such applications it is better to optimize the orbitals for the average energy of several electronic states. Further, convergence can be slower in some cases when orbitals in RAS1 and RAS3 are included. The program is not optimal for SDCI calculations with a large number of orbitals in RAS1 and RAS3.

It is best to provide a set of good input orbitals. (The program can be started from scratch by using CORE, but this should be used only if other possibilities fail). They can either be from some other type of calculation, for example `SCF`, or generated by `GUESSORB`, or from a previous `RASSCF` calculation on the same system. In the first case the orbitals are normally given in formatted form, file *`INPORB`*, in the second case they can also be read from a `RASSCF` input unit *`JOBOLD`*. Input provides both possibilities. Some care has to be taken in choosing the input orbitals, especially for the weakly occupied ones. Different choices may lead to convergence to different local minima. One should therefore make sure that the input orbitals have the correct general structure. A good strategy is often to start using a smaller basis set (MB or DZ) and once the orbitals have been defined, increase the basis set and use `EXPBAS` to generate input orbitals.

When we speak of files like *`INPORB`* or *`JOBIPH`*, please note that these can be regarded as generic names. You may have various files with different file names available, and when invocating the `RASSCF` program, these can be linked or copied (See EMIL command LINK and COPY) so that the program treats them as having the names INPORB or JOBIPH. Also, by the commands FILEORB and IPHNAME, you can instruct the program to use other file names.

There is two kinds of specifications to make for orbitals: One is the coefficient arrays that describe the molecular orbitals, commonly called 'CMO data'. The other kind is the number of inactive, ras1, etc. orbitals of each symmetry type, which will be called 'orbital specifications'. The program can take either or both kinds of data from *`INPORB`*, *`JOBIPH`* or runfile. The program selects where to fetch such data, based on rules and input keywords. Avoid using conflicting keywords: the program may sometimes go ahead and resolve the problem, or it may decide to stop, not to risk wasting the user's time on a calculation that was not intended. This decision may be in error.

The orbital specification by keyword input is easy: See keywords FROZEN, INACTIVE, etc. If any such keyword is used, then all the orbital specifications are assumed to be by input, and any such input that is lacking is determined by default rules. These are that there are no such orbitals, with the exception of DELETED: If earlier calculations deleted some orbitals for reason of (near) linear dependence, then these will continue being deleted in subsequent calculations, and cannot be 'undeleted'.

If no such keyword has been given, but keyword LUMO is used to instruct the program to fetch CMO data from *INPORB*, then also the orbitals specs are taken from *INPORB*, if (as is usually the case) this file contains so-called **typeindex** information. The `GV` program may have been used to graphically view orbital pictures and pick out suitable active orbitals, etc., producing a file with extension '.GvOrb'. When this is used as *INPORB* file, the selected orbitals will be picked in the correct order.

In case both keywords, such as INACTIVE, **and** LUMORB, is given, this is of course the very common case that CMO data are read from *INPORB* but orbital specs are given by input. This is perhaps the most common usage. However, when the *INPORB* file is a produced by `GV`, it happens frequently that also keyword specs are left in the input, since the user knows that these merely duplicate the specs in *INPORB*. But the latter may also imply a reordering of the orbitals. For this reason, when the keyword input merely duplicates the number of inactive, etc., that is also specified by typeindex, then the typeindex input overrides, to produce the correct ordering. If they do **not** match precisely, then the CMO data are read, without reordering, and the keyword input (as usual) takes precedence.

The CMO data are obtained as follows: With the following keywords, it is assumed that the user knows what he wants.

- CORE: (A bad choice, but here for completeness). Creates orbitals from scratch.
- LUMORB or FILEORB: Try *INPORB*, or fail.
- JOBIPH: Try *JOBOLD*, if not usable, try *JOBIPH*, or fail.

If none of these keywords were used, then the user accepts defaults, namely

1. look for rasscf orbitals on *RUNFILE*
2. look for scf orbitals on *RUNFILE*
3. look for guessorb orbitals on *RUNFILE*
4. If still nothing found, create orbitals from scratch.

As for earlier versions, notice the possibility to read the orbitals on *JOBIPH*, at a later time, by using the keywords OUTORBITAL and ORBONLY. This results in editable ASCII files, with names like Project.RasOrb (or Project.RasOrb5 for the fifth root). Such orbitals will be produced by default for the lowest roots – up to the tenth, named now, e.g., Project.RasOrb.5. There is a keyword MAXORB to produce more (or fewer) such files.

The `RASSCF` program has special input options, which will limit the degrees of freedoms used in the orbital rotations. It is, for example, possible to impose averaging of the orbital densities in π symmetries for linear molecules. Use the keyword AVERAGE for this purpose. It is also possible to prevent specific orbitals from rotating with each other. The keyword is SUPSYM. This can be used, for example, when the molecule has higher symmetry than one can use with the *MOLCAS* system. For example, in a linear molecule the point group to be used is C_{2v} or D_{2h}. Both $\sigma-$ and $\delta-$orbitals will then appear in irrep 1. If the input orbitals have been prepared to be adapted to linear symmetry, the SUPSYM input can be used to keep this symmetry through the iterations. The program will do this automatically with the use of the input keyword LINEAR. Similarly, for single atoms, spherical symmetry can be enforced by the keyword ATOM.

RASSCF output orbitals

The `RASSCF` program produces a binary output file called *`JOBIPH`*, which can be used in subsequent calculations. Previously, this was usually a link, pointing to whichever file the user wanted, or by default to the file *`$Project.JobIph`* if no such links had been made. This default can be changed, see keyword NEWIPH and IPHNAME. For simplicity, we refer to this as *`JOBIPH`* in the manual.The job interface, *`JOBIPH`*, contains four different sets of MO's and it is important to know the difference between the sets:

1. **Average orbitals:** These are the orbitals produced in the optimization procedure. Before performing the final CI wave function they are modified as follows: inactive and secondary orbitals are rotated (separately) such as to diagonalize an effective Fock operator, and they are then ordered after increasing energy. The orbitals in the different RAS subspaces are rotated (within each space separately) such that the corresponding block of the state-average density matrix becomes diagonal. These orbitals are therefore called "pseudo-natural orbitals". They become true natural orbitals only for CAS type wave functions. These orbitals are not ordered. The corresponding "occupation numbers" may therefore appear in the output in arbitrary order. The final CI wave function is computed using these orbitals. They are also the orbitals found in the printed output.

2. **Natural orbitals:** They differ from the above orbitals, in the active subspace. The entire first order density matrix has been diagonalized. Note that in a RAS calculation, such a rotation does not in general leave the RAS CI space invariant. One set of such orbitals is produced for each of the wave functions in an average `RASSCF` calculation. The main use of these orbitals is in the calculation of one-electron properties. They are extracted by default (up to ten roots) to the working directory from *`JOBIPH`* and named *`$Project.RasOrb.1`*, *`$Project.RasOrb.2`*, etc. Each set of MO's is stored together with the corresponding occupation numbers. The natural orbitals are identical to the average orbitals only for a single state CASSCF wave function.

3. **Canonical orbitals:** This is a special set of MO's generated for use in the `CASPT2` and `CCSDT` programs. They are obtained by a specific input option to the `RASSCF` program. They are identical to the above orbitals in the inactive and secondary subspaces. The active orbitals have been obtained by diagonalizing an effective one-electron Hamiltonian, a procedure that leaves the CI space invariant only for CAS type wave functions.

4. **Spin orbitals:** This set of orbitals is generated by diagonalizing the first order spin density matrix and can be used to compute spin properties.

8.30.1 Dependencies

To start the `RASSCF` module at least the one-electron and two-electron integrals generated by `SEWARD` have to be available (exception: See keyword ORBONLY). Moreover, the `RASSCF` requires a suitable start wave function such as the orbitals from a RHF-SCF calculation or produced by `GUESSORB`.

8.30.2 Files

Input files

File	*Contents*
ONEINT	Contains the one-electron integrals
*ORDINT**	Contains the two-electron integrals.
RUNFILE	The *RUNFILE* contains auxiliary information.
INPORB	This a formatted ASCII file including suitable start orbitals.
JOBOLD	The binary output file, called *JOBIPH*, which has been generated by the `RASSCF` program can be used under the name *JOBOLD* to start a new calculation.

A number of additional files generated by `SEWARD` are also used by the `RASSCF` program. The availability of either of the files named *INPORB* and *JOBOLD* is optional and determined by the input options LUMORB and JOBIPH, respectively.

Output files

File	*Contents*
JOBIPH	This file is written in binary format and carries the results of the wave function optimization such as MO- and CI-coefficients. If several consecutive RASSCF calculations are made, the file names will be modified by appending '01','02' etc.
RUNFILE	The *RUNFILE* is updated with information from the RASSCF calculation such as the first order density and the Fock matrix.
MOLDEN.x	Molden input file for molecular orbital analysis for CI root x.
RASORB	This ASCII file contains molecular orbitals, occupation numbers, and orbital indices from a `RASSCF` calculation. The natural orbitals of individual states in an average-state calculation are also produced, and are named RASORB.1, RASORB.2, etc.

8.30.3 Input

This section describes the input to the `RASSCF` program in the *MOLCAS* program system. The input starts with the program name

```
␣&RASSCF
```

There are no compulsory keywords, but almost any meaningful calculation will require some keyword. At the same time, most choices have default settings, and many are able to take relevant values from earlier calculations, from available orbital files, etc.

Optional keywords

There is a large number of optional keywords you can specify. They are used to specify the orbital spaces, the CI wave function etc., but also more arcane technical details that can modify e.g. the convergence or precision. The first 4 characters of the keyword are recognized by the input parser and the rest is ignored. If not otherwise stated the numerical input that follows a keyword is read in free format. A list of these keywords is given below:

Keyword	*Meaning*
TITLe	After this keyword follows a single line with a title for the calculation.
SYMMetry	Specify the selected symmetry type (the irrep) of the wave function as a number between 1 and 8 (see SEWARD). Default is 1, which always denote the totally symmetric irrep.
SPIN	The keyword is followed by an integer giving the value of spin multiplicity ($2S + 1$). Default is 1 (singlet).
CHARge	Specify the total charge on the system as an integer. If this keyword is used, the NACTEL keyword should not be used. Default value: 0
NACTel	A line with three numbers follows, specifying 1. the total number of active electrons (all electrons minus twice the number of inactive and frozen orbitals) 2. the maximum number of holes in Ras1 3. the maximum number of electrons occupying the Ras3 orbitals Default values are: x,0,0, where x is the number needed to get a neutral system. See also keyword CHARGE, which is an alternative specification.
CIROot	Specifies the CI root(s) and the dimension of the starting CI matrix used in the CI Davidson procedure. This input makes it possible to perform orbital optimization for the average energy of a number of states. The first line of input gives two or three numbers, specifying the number of roots used in the average calculation (NROOTS), the dimension of the small CI matrix in the Davidson procedure (LROOTS), and possibly a non-zero integer IALL. If $IALL.ne.1$ or

there is no IALL, the second line gives the index of the states over which the average is taken (NROOTS numbers, IROOT). **Note** that the size of the CI matrix, LROOTS, must be at least as large as the highest root, IROOT. If, **and only if**, NROOTS>1 a third line follows, specifying the weights of the different states in the average energy. If IALL=1 has been specified, no more lines are read. A state average calculation will be performed over the NROOTS lowest states with equal weights. energy. Examples:

```
CIRO
␣3␣5
␣2␣4␣5
␣1␣1␣3
```

The average is taken over three states corresponding to roots 2, 4, and 5 with weights 20%, 20%, and 60%, respectively. The size of the Davidson Hamiltonian is 5. Another example is:

```
CIRO
␣19␣19␣1
```

A state average calculation will be performed over the 19 lowest states each with the weight 1/19 Default values are NROOTS = LROOTS = IROOT=1 (ground state), which is the same as the input:

```
CIRO
␣1␣1
␣1
```

ATOM This keyword is used to get orbitals with pure spherical symmetry for atomic calculations (the radial dependence can vary for different irreps though). It causes super-symmetry to be switched on (see SUPSym keyword) and generates automatically the super-symmetry vector needed. Also, at start and after each iteration, it sets to zero any CMO coefficients with the wrong symmetry. Use this keyword instead of SUPSym for atoms.

LINEar This keyword is used to get orbitals with pure rotational symmetry for linear molecules. It causes super-symmetry to be switched on (see SUPSym keyword) and generates automatically the super-symmetry vector needed. Also, at start and after each iteration, it sets to zero any CMO coefficients with the wrong symmetry. Use this keyword instead of SUPSym for linear molecules.

RLXRoot Specifies which root to be relaxed in a geometry optimization of a state average wave function. Thus, the key word has to be combined with CIRO. In a geometry optimization the following input

```
CIRO
␣3␣5
␣2␣4␣5
␣1␣1␣3
RLXR
4
```

will relax CI root number 4.

RFPErt	This keyword will add a constant reaction field perturbation to the Hamiltonian. The perturbation is read from the file ***RUNFILE*** and is the latest self-consistent perturbation generated by one of the programs `SCF` or `RASSCF`.
NONEquilibrium	Makes the slow components of the reaction field of another state present in the reaction field calculation (so-called non-equilibrium solvation). The slow component is always generated and stored on file for equilibrium solvation calculations so that it potentially can be used in subsequent non-equilibrium calculations on other states.
RFROot	Enter the index of that particular root in a state-average calculation for which the reaction-field is generated. It is used with the PCM model.
NEWIph	The default name of the JOBIPH file will be determined by any already existing such files in the work directory, by appending '01', '02' etc. so a new unique name is obtained.
IPHName	Override the default choice of name of the JOBIPH file by giving the file name you want.
FILEorb	Override the default name (INPORB) for starting orbital file by giving the file name you want.
FROZen	Specifies the number of frozen orbitals in each symmetry. (see below for condition on input orbitals). Frozen orbitals will not be modified in the calculation. Only doubly occupied orbitals can be left frozen. This input can be used for example for inner shells of heavy atoms to reduce the basis set superposition error. Default is 0 in all symmetries.
INACtive	Specify on the next line the number of inactive (doubly occupied) orbitals in each symmetry. Frozen orbitals should not be included here. Default is 0 in all symmetries.
RAS1	On the next line specify the number of orbitals in each symmetry for the Ras1 orbital subspace. Default is 0 in all symmetries.
RAS2	On the next line specify the number of orbitals in each symmetry for the Ras2 orbital subspace. Default is 0 in all symmetries.
RAS3	On the next line specify the number of orbitals in each symmetry for the Ras3 orbital subspace. Default is 0 in all symmetries.
DELEted	On the next line specify the number of deleted orbitals in each symmetry. These orbitals will not be allowed to mix into the occupied orbitals. It is always the last orbitals in each symmetry that are deleted. Default is 0 in all symmetries.
LUMOrb	Input molecular orbitals are read from a formatted file with FORTRAN file name ***INPORB***. **Note**, the keywords Lumorb, Core, and Jobiph are mutually exclusive. If none is given the program will

search for input orbitals on the runfile in the order: `RASSCF`, `SCF`, `GUESSORB`. If none is found, the keyword CORE will be activated.

JOBIPH Input molecular orbitals are read from an unformatted file with FORTRAN file name *`JOBOLD`*. **Note**, the keywords LUMORB, CORE, and JOBIPH are mutually exclusive. If none is given the program will search for input orbitals on the runfile in the order: `RASSCF`, `SCF`, `GUESSORB`. If none is found, the keyword CORE will be activated.

CORE Input molecular orbitals are obtained by diagonalizing the core Hamiltonian. This option is only recommended in simple cases. It often leads to divergence. **Note**, the keywords LUMORB, CORE, and JOBIPH are mutually exclusive.

TYPEINDEX This keyword forces the program to use information about subspaces from the INPORB file.

User can change the order of orbitals by editing of "Type Index" section in the INPORB file. The legend of the types is:

- **F** - Frozen
- **I** - Inactive
- **1** - RAS1
- **2** - RAS2
- **3** - RAS3
- **S** - Secondary
- **D** - Deleted

ALTER This keyword is used to change the ordering of MO in *`INPORB`* or *`JOBOLD`*. Next line must contain the number of pairs to be interchanged and it is followed by lines that specify, for each pair, the symmetry species of the pair and the indices of the two permuting MOs. Here is an example:

```
ALTEr
␣2
␣1␣4␣5
␣3␣6␣8
```

In this example, 2 pairs of MO will be exchanged: 4 and 5 in symmetry 1 and 6 and 8 in symmetry 3.

CLEANUP This input is used to set to zero specific coefficients of the input orbitals. It is of value, for example, when the actual symmetry is higher than given by input and the trial orbitals are contaminated by lower symmetry mixing. The input requires at least one line per symmetry specifying the number of additional groups of orbitals to clean. For each group of orbitals within the symmetry, three lines follow. The first line indicates the number of considered orbitals and the specific number of the orbital (within the symmetry) in the set of input orbitals. Note the input lines can not be longer than 72 characters and the program expects as many continuation lines as are needed. The second line indicates the number of coefficients belonging to the prior orbitals which are going to be set to zero and which coefficients. The third line indicates the number of the coefficients of all the complementary orbitals of the symmetry which are going to be set to zero and which are these coefficients. Here is an example of what an input would look like:

```
CLEAnup
2
␣␣␣3␣␣4␣7␣9
␣␣␣3␣␣10␣11␣13
␣␣␣4␣␣12␣15␣16␣17
␣␣␣2␣␣8␣11
␣␣␣1␣␣15
␣␣␣0
0
0
0
```

In this example the first line indicates that two groups of orbitals are specified in the first symmetry. The first item of the following line indicates that there are three orbitals considered (4, 7, and 9). The first item of the following line indicates that there are three coefficients of the orbitals 4, 7, and 9 to be set to zero, coefficients 10, 11, and 13. The first item of the following line indicates that there are four coefficients (12, 15, 16, and 17) which will be zero in all the remaining orbitals of the symmetry. For the second group of the first symmetry orbitals 8 and 11 will have their coefficient 15 set to zero, while nothing will be applied in the remaining orbitals. If a geometry optimization is performed the keyword is inactive after the first structure iteration.

CIREstart

Starting CI-coefficients are read from a binary file *JOBOLD*.

ORBOnly

This input keyword is used to get a formated ASCII file (RASORB, RASORB.2, etc) containing molecular orbitals and occupations reading from a binary JobIph file. The program will not perform any other operation. (In this usage, the program can be run without any files, except the JOBIPH file).

CIONly

This keyword is used to disable orbital optimization, that is, the CI roots are computed only for a given set of input orbitals.

CHOInput

This marks the start of an input section for modifying the default settings of the Cholesky RASSCF. Below follows a description of the associated options. The options may be given in any order, and they are all optional except for **ENDChoinput** which marks the end of the **CHOInput** section.

- **NoLK** - Available only within ChoInput. Deactivates the "Local Exchange" (LK) screening algorithm [121] in computing the Fock matrix. The loss of speed compared to the default algorithm can be substantial, especially for electron-rich systems. Default is to use LK.

- **DMPK** - Available only within ChoInput. Modifies the thresholds used in the LK screening. The keyword takes as argument a (double precision) floating point (non-negative) number used as correction factor for the LK screening thresholds. The default value is 1.0d-1. A smaller value results in a slower but more accurate calculation.
  Note.: The default choice of the LK screening thresholds is tailored to achieve as much as possible an accuracy of the converged RASSCF energies consistent with the choice of the Cholesky decomposition threshold.
- **NODEcomposition** - Available only within ChoInput. Deactivates the Cholesky decomposition of the inactive AO 1-particle density matrix. The inactive Exchange contribution to the Fock matrix is therefore computed using inactive canonical orbitals instead of (localized) “Cholesky MOs” [111]. This choice tends to lower the performances of the LK screening. Default is to perform this decomposition in order to obtain the Cholesky MOs.
- **TIME** - Activates printing of the timings of each task of the Fock matrix build. Default is to not show these timings.
- **MEMFraction** - Set the fraction of memory to use as global Cholesky vector buffer. Default: for serial runs 0.0d0; for parallel runs 0.3d0.

ITERations	Specify the maximum number of `RASSCF` iterations, and the maximum number of iterations used in the orbital optimization (super-CI) section. Default and maximum values are 200,100.
LEVShft	Define a level shift value for the super-CI Hamiltonian. Typical values are in the range 0.0 – 1.5. Increase this value if a calculation diverges. The default value 0.5, is normally the best choice when Quasi-Newton is performed.
THRS	Specify convergence thresholds for: energy, orbital rotation matrix, and energy gradient. Default values are: 1.0e-08, 1.0e-04, 1.0e-04.
TIGHt	Convergence thresholds for the Davidson diagonalization procedure. Two numbers should be given: THREN and THFACT. THREN specifies the energy threshold in the first iteration. THFACT is used to compute the threshold in subsequent iterations as THFACT*DE, where DE is the RASSCF energy change. Default values are 1.0d-04 and 1.0d-3.
NOQUne	This input keyword is used to switch off the Quasi-Newton update procedure for the Hessian. Pure super-CI iterations will be performed. (Default setting: QN update is used unless the calculation involves numerically integrated DFT contributions.)
QUNE	This input keyword is used to switch on the Quasi-Newton update procedure for the Hessian. (Default setting: QN update is used unless the calculation involves numerically integrated DFT contributions.)

CIMX

Specify the maximum number of iterations allowed in the CI procedure. Default is 100 with maximum value 200.

SDAV

Here follows the dimension of the explicit Hamiltonian used to speed up the Davidson CI iteration process. An explicit H matrix is constructed for those configurations that have the lowest diagonal elements. This H-matrix is used instead of the corresponding diagonal elements in the Davidson update vector construction. The result is a large saving in the number if CI iterations needed. Default value is the smallest of 100 and the number of configurations. Increase this value if there is problems converging to the right roots.

SXDAmp

A variable called SXDAMP regulates the size of the orbital rotations. Use keyword SXDAmp and enter a real number on the next line. The default value is 0.0002. Larger values can give slow convergence, very low values may give problems e.g. if some active occupations are very close to 0 or 2.

SUPSym

This input is used to restrict possible orbital rotations. It is of value, for example, when the actual symmetry is higher than given by input. Each orbital is given a label IXSYM(I). If two orbitals in the same symmetry have different labels they will not be allowed to rotate into each other and thus prevents from obtaining symmetry broken solutions. Note, however, that the starting orbitals must have the right symmetry. The input requires one or more lines per symmetry. The first specifies the number of additional subgroups in this symmetry (a 0 (zero) denotes that there is no additional subgroups and the value of IXSYM will be 0 (zero) for all orbitals in that symmetry). If the number of additional subgroups is not zero there are (an) additional line(s) for each subgroup: The dimension of the subgroup and the list of orbitals in the subgroup counted relative to the first orbital in this symmetry. Note, the input lines can not be longer than 72 characters and the program expects continuation lines as many as there are needed. As an example assume an atom treated in C_{2v} symmetry for which the d$_{z^2}$ orbitals (7 and 10) in symmetries 1 may mix with the s orbitals. In addition, the d$_{z^2}$ and d$_{x^2-y^2}$ orbitals (8 and 11) may also mix. Then the input would look like:

```
SUPSym
2
␣␣␣2␣␣7␣10
␣␣␣2␣␣8␣11
0
0
0
```

In this example the first line indicates that we would like to specify two additional subgroups in the first symmetry (total symmetric group). The first item in the following two lines declares that each subgroup consists of two orbitals. Orbitals 7 and 10 constitute the first group and it is assumed that these are orbitals of d$_{z^2}$ character. The second group includes the d$_{x^2-y^2}$ orbitals 8 and 11. The following three lines denote that there are no further subgroups defined for the remaining symmetries. Ordering of the orbitals according to energy is deactivated when using SUPSym. If you activate ordering using ORDEr, the new labels will be printed in the output section. If a geometry optimization is performed the reordered matrix will be stored in the ***`RUNFILE`*** file and read from there instead of from the input in each new structure iteration.

HOME With this keyword, the root selection in the Super-CI orbital update is by maximum overlap rather than lowest energy.

VB Using this keyword, the CI optimization step in the RASSCF program will be replaced by a call to the *CASVB* program, such that fully variational valence bond calculations may be carried out. The `VB` keyword can be followed by any of the directives described in section 8.3 and should be terminated by `ENDVB`. Energy-based optimization of the VB parameters is the default, and the output level for the main *CASVB* iterations is reduced to -1, unless the print level for *CASVB* print option 6 is $\geq$2.

PRINt The keyword is followed by a line giving the print levels for various logical code sections. It has the following structure: IW IPR IPRSEC(I), I=1,7

- IW - logical unit number of printed output (normally 6).
- IPR - is the overall print level (normally 2).
- IPRSEC(I) - gives print levels in different sections of the program.
    1. Input section
    2. Transformation section
    3. CI section
    4. Super-CI section
    5. Not used
    6. Output section
    7. Population analysis section

The meaning of the numbers: 0=Silent, 1=Terse, 2=Normal, 3=Verbose, 4=Debug, and 5=Insane. If input is not given, the default (normally=2) is determined by a global setting which can be altered bubroutine call. (Programmers: See programming guide). The local print level in any section is the maximum of the IPRGLB and IPRSEC() setting, and is automatically reduced e.g. during structure optimizations or numerical differentiation. Example: 6 2 2 2 3 2 2 2 2

MAXOrb Max nr of RasOrb files to produce, one for each root up to the maximum.

OUTOrbitals This keyword is used to select the type of orbitals to be written in a formated ASCII file. By default a formated ***`RASORB`*** file containing average orbitals and subsequent ***`RASORB.1`***, ***`RASORB.2`***, etc, files containing natural orbitals for each of the computed (up to ten) roots will be generated in the working directory. A line follows with an additional keyword selecting the orbital type. The possibilities are:

AVERage orbitals: this is the default option. This keyword is used to produce a formated ASCII file of orbitals (RASORB) which correspond to the final state average density matrix obtained by the `RASSCF` program. The inactive and secondary orbitals have been transformed to make an effective Fock matrix diagonal. Corresponding diagonal elements are given as orbital energies in the `RASSCF` output listing. The active orbitals have been obtained by diagonalizing the sub-blocks of the average density matrix corresponding to the three different RAS orbital spaces, thereby the name pseudo-natural orbitals. They will be true natural orbitals only for a CASSCF wave function.

CANOnical orbitals: Use this keyword to produce the canonical orbitals. They differ from the natural orbitals, because also the active part of the Fock matrix is diagonalized. Note that the density matrix is no longer diagonal and the CI coefficients have not been transformed to this basis. This option substitutes the previous keyword CANOnical.

NATUral orbitals: Use this keyword to produce the true natural orbitals. The keyword should be followed by a new line with an integer specifying the maximum CI root for which the orbitals and occupation numbers are needed. One file for each root will be generated up to the specified number. In a one state RASSCF calculation this number is always 1, but if an average calculation has been performed, the NO's can be obtained for all the states included in the energy averaging. Note that the natural orbitals main use is as input for property calculations using `SEWARD`. The files will be named RASORB, RASORB.2, RASORB.3, etc. This keyword is on by default for up to ten roots.

SPIN orbitals. This keyword is used to produce a set of spin orbitals and is followed by a new line with an integer specifying the maximum CI root for which the orbitals and occupation numbers are needed. One file for each root will be generated up to the specified number. Note, for convenience the doubly occupied and secondary orbitals have been added to these sets with occupation numbers 0 (zero). The main use of these orbitals is to act as input to property calculations and for graphical presentations. This keyword is on by default for up to ten roots.

An example input follows in which five files are requested containing natural orbitals for roots one to five of a RASSCF calculation. The files are named RASORB.1, RASORB.2, RASORB.3, RASORB.4, and RASORB.5, respectively for each one of the roots. Although this is the default, it can be used complemented by the ORBOnly keyword, and the orbitals will be read from a JobIph file from a previous calculation, in case the formated files were lost or you require more than ten roots. As an option the MAXOrb can be also used to increase the number of files over ten.

```
OUTOrbitals
␣Natural␣Orbitals
␣15
```

ORBLISTING

This keyword is followed on the next line with a word showing how extensive you want the orbital listing to be in the printed output. The alternatives are:

- **NOTHing:** No orbitals will be printed (suggested for numerical CASPT2 optimization). (Also, the old VERYbrief will be accepted).
- **FEW:** The program will print the occupied orbitals, and any secondary with less than 0.15 a.u. orbital energy. (Old BRIEF also accepted).
- **NOCOre:** The program will print the active orbitals, and any secondary with less than 0.15 a.u. orbital energy.
- **ALL:** All orbitals are printed. (Old LONG also accepted).

(unless other limits are specified by the PROR keyword).

ORBAPPEAR

This keyword is followed on the next line with a word showing the appearance of the orbital listing in the printed output. The alternatives are:

- **COMPact:** The format of the orbital output is changed from a tabular form to a list giving the orbital indices and MO-coefficients. Coefficients smaller than 0.1 will be omitted.
- **FULL:** The tabular form will be chosen.

PROR

This keyword is used to alter the printout of the MO-coefficients. Two numbers must be given of which the first is an upper boundary for the orbital energies and the second is a lower boundary for the occupation numbers. Orb¡/KEYWORD¿itals with energy higher than the threshold or occupation numbers lower that the threshold will not be printed. By default these values are set such that all occupied orbitals are printed, and virtual orbitals with energy less than 0.15 au. However, the values are also affected by use of OUTPUT.

ORDER

This input keyword is used to deactivate or activate ordering of the output orbitals according to energy. One number must be given on the line following the keyword, 1 if you want ordering and 0 if you want to deactivate ordering. Default is 1 and with SUPSym keyword default is 0.

PRSP

Use this keyword to get the spin density matrix for the active orbitals printed.

PRWF

Enter the threshold for CI coefficients to be printed (Default: 0.05).

A general comment concerning the input orbitals: The orbitals are ordered by symmetry. Within each symmetry block the order is assumed to be: frozen, inactive, active, external (secondary), and deleted. Note that if the SPDELETE option has been used in a preceding `SCF` calculation, the deleted orbitals will automatically be placed as the last ones in each symmetry block.

For calculations of a molecule in a reaction field see section 8.14.1 of the present manual and section 5.6 of the examples manual.

Input example

The following example shows the input to the `RASSCF` program for a calculation on the water molecule. The calculation is performed in C_{2v} symmetry (symmetries: a_1, b_2, b_1, a_2, where the two last species are antisymmetric with respect to the molecular plane). Inactive orbitals are $1a_1$ (oxygen $1s$) $2a_1$ (oxygen 2s) and $1b_1$ (the π lone-pair orbital). Two bonding and two anti-bonding OH orbitals are active, a_1 and b_2 symmetries. The calculation is performed for the ${}^1A_1$ ground state. Note that no information about basis set, geometry, etc has to be given. Such information is supplied by the `SEWARD` integral program via the one-electron integral file *`ONEINT`*.

```
␣&RASSCF
Title
␣Water␣molecule.␣Active␣orbitals␣OH␣and␣OH*␣in␣both␣symmetries
Spin
␣1
Symmetry
␣1
Inactive
␣2␣0␣1␣0
Ras2
␣2␣2␣0␣0
```

More advanced examples can be found in the tutorial section of the manual.

8.31 `RASSI`

The `RASSI` (RAS State Interaction) program forms overlaps and other matrix elements of the Hamiltonian and other operators over a wave function basis, which consists of RASSCF wave functions, each with an individual set of orbitals. It is extensively used for computing dipole oscillator strengths, but any one-electron operator, for which the `SEWARD` has computed integrals to the *`ORDINT`* file, can be used, not just dipole moment components.

Also, it solves the Schrödinger equation projected on the space spanned by these wave functions, i.e., it forms non-interacting linear combinations of the input state functions, and computes matrix elements over the resulting eigenbasis as well.

Finally, using these spin-free eigenstates as a basis, it can compute spin-orbit interaction matrix elements, diagonalize the resulting matrix, and compute various matrix elements over the resulting set of spin-orbit eigenstates.

If only matrix elements of some one-electron operator(s), such as the dipole transition moments, are required, the calculation of Hamiltonian matrix elements and the transformation to the eigenbasis of this matrix can be skipped. However, if any states have the same symmetry and different orbitals, it is desirable to use the transitions strengths as computed between properly non-interacting and orthonormal states. The reason is that the individually optimized RASSCF states are interacting and non-orthogonal, and the main error in the computed transition matrix elements is the difference in electronic dipole moment times the overlap of any two states involved. For excited states, the overlap is often in the order of 10%.

Please note: Due to the increasing number of calculations done with a hundred input states, or more, there has been a demand to change the output. Until Molcas 6.2, the default assumption has been to print all expectation values and matrix elements that can be computed from the selection of one-electron integrals. From 6.4, this is requested by keywords, see the keyword list below for XVIN, XVES, XVSO, MEIN, MEES, and MESO.

Apart from computing oscillator strengths, overlaps and Hamiltonian matrix elements can be used to compute electron transfer rates, or to form quasi-diabatic states and reexpress matrix elements over a basis of such states.

The CSF space of a RASSCF wave function is closed under deexcitation. For any given pair of RASSCF wave functions, this is used in the way described in reference [122] to allow the pair of orbital sets to be transformed to a biorthonormal pair, while simultaneously transforming the CI expansion coefficients so that the wave functions remain unchanged. The basic principles are the same as in the earlier program [123], but is adapted to allow RASSCF as well as CASSCF wave functions. It uses internally a Slater determinant expansion. It can now use spin-dependent operators, including the AMFI spin-orbit operator, and can compute matrix elements over spin-orbit states, i.e. the eigenstates of the sum of the spin-free hamiltonian and the spin-orbit operator.

One use of the RASSI eigenstates is to resolve ambiguities due to the imperfect description of highly excited states. Association between individually optimized states and the exact electronic eigenstates is often not clear, when the calculation involves several or many excited states. The reason is that the different states each use a different set of orbitals. The State Interaction calculation gives an unambiguous set of non-interacting and orthonormal eigenstates to the projected Schrödinger equation, and also the overlaps between the original RASSCF wave functions and the eigenstates. The latter is a very efficient diagnostic, since it describes the RASSCF states in terms of one single wave-function basis set.

To make the last point clear, assume the following situation: We have performed three RASSCF calculations, one where we optimize for the lowest state, one for the first excited state, and one for the 2nd excited state in the same symmetry. The active orbitals are fairly much mixed around, so a simple inspection of the CI coefficient is insufficient for comparing the states. Assume that for each state, we have calculated the three lowest CI roots. It can now happen, that the 2nd root of each calculation is a fair approximation to the exact 2nd eigenstate, and the same with the 3rd, or possibly that the order gets interchanged in one or two of the calculation. In that case, a RASSI calculation with these 9 states will give three improved solutions close to the original ones, and of course 6 more that are considered to be the removed garbage. The overlaps will confirm that each of the input states consists mainly of one particular out of the three lowest eigenstates. This situation is the one we usually assume, if no further information is available.

However, it happens that the active orbitals of the three calculations do not span approximately the same space. The orbital optimization procedure has made a qualitatively different selection of correlating orbitals for the three different calculation. Then the `RASSI` calculation may well come out with 4 lowest roots that overlap strongly with the original RASSCF states. This may change the assignments and may also give valuable information about the importance of some state. The natural orbitals of the eigenstates will show that the active space used in the RASSCF was in some way inappropriate.

Another bothersome situation is also solved by the RASSI method. The analysis of the original states in terms of RASSI eigenstates may reveal that the three optimized RASSCF states consists mainly of TWO low RASSI eigenstates! This is because the RASSCF optimization equations are non-linear and may sometimes offer spurious extra solutions. Two of the calculations are in this case to be regarded qualitatively, as two different (local) solutions that approximate (imperfectly) the same excited state. Also in this case, the natural orbitals will probably offer a clue to how to get rid of the problem. Extra solutions rarely occur for low states in CASSCF calculations, provided a generous active space can be afforded. Problems occur when the active space is too small, and in particular with general RASSCF calculations.

A further application is the preparation of a suitable orbital basis for a subsequent CI calculation. Note that such an application also allows the use of badly converged RASSCF wave functions, or of RASSCF wave functions containing multiple minima solutions close to a common exact eigenstate. In effect, the RASSI program cleans up the situation by removing the errors due to bad convergence (pushing the errors into a garbage part of the spectrum). This requires that the set of input states (9 in this example) provides flexibility enough to remove at least a major part of the error. As one would expect, this is usually true: The erratic non-convergent, or the too slowly convergent, error mode is to a large extent spanned by the few lowest RASSCF wave functions.

Finally, there are situations where there is no problem to obtain adiabatic RASSCF solutions, but where it is still imperative to use RASSI natural orbitals in a subsequent CI. Consider the case of transition metal chemistry, where there is in general two or more electronic states involved. These states are supposed to interact strongly, at least within some range of interatomic distances. Here, an MCSCF solution, such as RASSCF, will have at least two very different solutions, one associated with each configuration of the transition metal atom. Using one set of orbitals, one electronic state has a reasonably described potential energy curve, while other states get pushed far up in energy. Using another set of orbitals, another state gets correctly described. In no calculation with a single orbital set do we obtain the avoided crossings, where one switches from one diabatic state to another. The only way to accomplish this is via a RASSI calculation. In this case, it is probably necessary also to shift the energies of the RASSCF states to ensure that the crossing occur at the correct places. The shifts can be determined by correcting the atomic spectrum in the separated-atoms limit.

Note, however, that most of the problems described above can be solved by performing state-averaged RASSCF calculations.

8.31.1 Dependencies

The `RASSI` program needs one or more JOBIPH files produced by the `RASSCF` program. Also, it needs a ONEINT file from `SEWARD`, with overlap integrals and any one-electron property integrals for the requested matrix elements. If Hamiltonian matrix elements are used, also the ORDINT file is needed.

8.31.2 Files

Like all the MOLCAS programs, RASSI opens the RUNFILE.

Input files

File	*Contents*
*ORDINT**	Ordered two-electron integral file produced by the `SEWARD` program. In reality, this is up to 10 files in a multi-file system, named ORDINT, ORDINT1,...,ORDINT9. This is necessary on some platforms in order to store large amounts of data.
ONEINT	The one-electron integral file from `SEWARD`
JOBnnn	A number of *JOBIPH* files from different `RASSCF` jobs. An older naming convention assumes file names JOB001, JOB002 etc for these files. They are automatically linked to default files named $Project.JobIph, $Project.JobIph01, $Project.JobIph02 etc. in directory $WorkDir, unless they already exists as files or links before the program starts. You can set up such links yourself, or else you can specify file names to use by the keyword FILES.
JOBIPHnn	A number of *JOBIPH* files from different `RASSCF` jobs. The present naming convention assumes file names JOBIPH, JOBIPH01 etc for such files, when created by subsequent `RASSCF` runs, unless other names were specified by input. They are automatically linked to default files named $Project.JobIph, $Project.JobIph01, $Project.JobIph02 etc. in directory $WorkDir, unless they already exists as files or links before the program starts. You can set up such links yourself, or else you can specify file names to use by the keyword FILES.

Output files

File	*Contents*
SIORBnn	A number of files containing natural orbitals, (numbered sequentially as *SIORB01*, *SIORB02*, etc.) If you do not like these names, you must set up links yourselves, except for the first one: SIORB01 is automatically linked to a default file named $Project.SiOrb in directory $WorkDir, if it does not already exists as a file or a link before the program starts. You must set up the other files yourself.
TOFILE	This output is only created if TOFILE is given in the input. It will contain the transition density matrix computed by `RASSI`. Currently, this file is only used as input to `QMSTAT`.

EIGV	Like *TOFILE* this file is only created if TOFILE is given in the input. It contains auxiliary information that is picked up by `QMSTAT`.

8.31.3 Input

This section describes the input to the `RASSI` program in the *MOLCAS* program system, with the program name:

```
 &RASSI
```

When a keyword is followed by additional mandatory lines of input, this sequence cannot be interrupted by a comment line. The first 4 characters of keywords are decoded. An unidentified keyword makes the program stop.

Keywords

Keyword	*Meaning*
CHOINPUT	This marks the start of an input section for modifying the default settings of the Cholesky RASSI. Below follows a description of the associated options. The options may be given in any order, and they are all optional except for **ENDChoinput** which marks the end of the **CHOInput** section. • **NoLK** - Available only within ChoInput. Deactivates the "Local Exchange" (LK) screening algorithm [121] in computing the Fock matrix. The loss of speed compared to the default algorithm can be substantial, especially for electron-rich systems. Default is to use LK. • **DMPK** - Available only within ChoInput. Modifies the thresholds used in the LK screening. The keyword takes as argument a (double precision) floating point (non-negative) number used as correction factor for the LK screening thresholds. The default value is 1.0d-1. A smaller value results in a slower but more accurate calculation. **Note:** the default choice of the LK screening thresholds is tailored to achieve as much as possible an accuracy of the RASSI energies consistent with the choice of the Cholesky decomposition threshold. • **NODEcomposition** - Available only within ChoInput. The inactive Exchange contribution to the Fock matrix is computed using inactive canonical orbitals instead of (localized) "Cholesky MOs". This choice is effective only in combination with the LK screening. Default is to use Cholesky MOs. **Note:** the Cholesky MOs in RASSI are computed by decomposing the

density type supermatrix $\mathbf{D} = (\mathbf{C}_A, \mathbf{C}_B)(\mathbf{C}_A, \mathbf{C}_B)^T$ where $\mathbf{C}$ is the corresponding canonical MOs matrix for the state A and B. When computing the coupling between 2 different states A and B, only for the first state we use pure Cholesky MOs. The invariance of the Fock matrix is then ensured by rotating the orbitals of B according to the orthogonal matrix defined in A through the Cholesky localization. These orbitals used for B are therefore called "pseudo Cholesky MOs".

- **TIME** - Activates printing of the timings of each task of the Fock matrix build. Default is to not show these timings.
- **MEMFraction** - Set the fraction of memory to use as global Cholesky vector buffer. Default: for serial runs 0.0d0; for parallel runs 0.3d0.

MEIN	Demand for printing matrix elements of all selected one-electron properties, over the input RASSCF wave functions.
MEES	Demand for printing matrix elements of all selected one-electron properties, over the spin-free eigenstates.
MESO	Demand for printing matrix elements of all selected one-electron properties, over the spin-orbit states.
PROPerty	Replace the default selection of one-electron operators, for which matrix elements and expectation values are to be calculated, with a user-supplied list of operators. From the lines following the keyword the selection list is read by the following *FORTRAN* code: READ(*,*) NPROP,(PNAME(I),ICOMP(I),I=1,NPROP) NPROP is the number of selected properties, PNAME(I) is a character string with the label of this operator on `SEWARD`'s one-electron integral file, and ICOMP(I) is the component number. The default selection is to use dipole and/or velocity integrals, if these are available in the ONEINT file. This choice is replaced by the user-specified choice if the PROP keyword is used. Note that the character strings are read using list directed input and thus must be within single quotes, see sample input below. For a listing of presently available operators, their labels, and component conventions, see `SEWARD` program description.
SOCOupling	Enter a positive threshold value. Spin-orbit interaction matrix elements over the spin components of the spin-free eigenstates will be printed, unless smaller than this threshold. The value is given in cm-1 units. The keyword is ignored unless an SO hamiltonian is actually computed.
SOPROPerty	Enter a user-supplied selection of one-electron operators, for which matrix elements and expectation values are to be calculated over the of spin-orbital eigenstates. This keyword has no effect unless the SPIN keyword has been used. Format: see PROP keyword.

SPINORBIT	Spin-orbit interaction matrix elements will be computed. Provided that the ONEL keyword was not used, the resulting Hamiltonian including the spin-orbit coupling, over a basis consisting of all the spin components of wave functions constructed using the spin-free eigenstates, will be diagonalized. NB: For this keyword to have any effect, the SO integrals must have been computed by SEWARD! See AMFI keyword in SEWARD documentation.
ONEL	The two-electron integral file will not be accessed. No Hamiltonian matrix elements will be calculated, and only matrix elements for the original RASSCF wave functions will be calculated.
J-VALUE	For spin-orbit calculations with single atoms, only: The output lines with energy for each spin-orbit state will be annotated with the approximate J and Omega quantum numbers.
OMEGA	For spin-orbit calculations with linear molecules, only: The output lines with energy for each spin-orbit state will be annotated with the approximate Omega quantum number.
IPHNAMES	Followed by one line for each JOBIPH file to be used, with the name of each file. Note: This keyword presumes that the number of jobiph files have already been entered using keyword NROF. For default *JOBIPH* file names, see the Files section.
NROF JOBIPHS	Number of *JOBIPH* files used as input. This keyword should be followed by the number of states to be read from each *JOBIPH*. Further, one line per *JOBIPH* is required with a list of the states to be read from the particular file. See sample input below. For *JOBIPH* file names, see the Files section.
SHIFT	The next line or lines gives an energy shift for each wave function, to be added to diagonal elements of the Hamiltonian matrix. This may be necessary e.g. to ensure that an energy crossing occurs where it should. NOTE: The number of states must be known (See keyword NROF) before this input is read. In case the states are not orthonormal, the actual quantity added to the Hamiltonian is 0.5D0*(ESHFT(I)+ESHFT(J))*OVLP(I,J). This is necessary to ensure that the shift does not introduce artificial interactions. SHIFT and HDIAG can be used together.
HDIAG	The next line or lines gives an energy for each wave function, to replace the diagonal elements of the Hamiltonian matrix. Non-orthogonality is handled similarly as for the SHIFT keyword. SHIFT and HDIAG can be used together.
NATORB	The next line gives the number of eigenstates, for which natural orbitals will be computed. They will be written, formatted, commented, and followed by natural occupancy numbers, on one file each. For file names, see the Files section. The format allows their use as standard orbital input files to other *MOLCAS* programs.
ORBITALS	Print out the Molecular Orbitals read from each *JOBIPH* file.

OVERLAPS	Print out the overlap integrals between the various orbital sets.
CIPRINT	Print out the CI coefficients read from *`JOBIPH`*.
THRS	The next line gives the threshold for printing CI coefficients. The default value is 0.05.
RFPE	RASSI will read from the runfile a response field contribution and add it to the Fock matrix.
HZER	The spin-free Hamiltonian is set to zero instead of being computed.
HEXT	It is read from the following few lines, as a triangular matrix: One element of the first row, two from the next, etc, as list-directed input of reals.
HEFF	A spin-free effective Hamiltonian is read from JOBIPH instead of being computed. It must have been computed by an earlier program. Presently, this is done by a multistate calculation using CASPT2. In the future, other programs may add dynamic correlation estimates in a similar way.
EJOB	The spin-free effective Hamiltonian is assumed to be diagonal, with energies being read from a JOBMIX file from a multistate CASPT2 calculation. In the future, other programs may add dynamic correlation estimates in a similar way.
TOFILE	Signals that a set of files with data from `RASSI` should be created. This keyword is necessary if `QMSTAT` is to be run afterwards.
XVIN	Demand for printing expectation values of all selected one-electron properties, for the input RASSCF wave functions.
XVES	Demand for printing expectation values of all selected one-electron properties, for the spin-free eigenstates.
XVSO	Demand for printing expectation values of all selected one-electron properties, for the spin-orbit states.

Input example

¿¿COPY ”Jobiph file 1” JOB001 ¿¿COPY ”Jobiph file 2” JOB002 ¿¿COPY ”Jobiph file 3” JOB003

```
␣&RASSI
NR␣OF␣JOBIPHS
␣␣␣␣3␣␣␣␣␣4␣␣␣2␣␣␣2␣␣␣␣--␣3␣JOBIPHs.␣Nr␣of␣states␣from␣each.
␣␣␣␣␣␣1␣␣␣2␣␣␣3␣␣␣4␣␣␣␣--␣Which␣roots␣from␣1st␣JOBIPH.
␣␣␣␣␣␣3␣␣␣4
␣␣␣␣␣␣3␣␣␣4
CIPR
THRS
␣␣␣0.02
Properties
␣␣␣␣4
```

```
'MltPl␣␣1'␣␣1␣␣␣'MltPl␣␣1'␣␣3␣␣␣␣'Velocity'␣␣1
'Velocity'␣␣3
*␣This␣input␣will␣compute␣eigenstates␣in␣the␣space
*␣spanned␣by␣the␣8␣input␣functions.␣Assume␣only␣the␣first
*␣4␣are␣of␣interest,␣and␣we␣want␣natural␣orbitals␣out
NATO
␣␣␣␣4
```

8.32 SCF

8.32.1 Description

The SCF program of the *MOLCAS* program system generates closed-shell Hartree-Fock, open-shell UHF, and Kohn Sham DFT wave functions.

The construction of the Fock matrices is either done conventionally from the two-electron integral file *ORDINT*, which was generated in a previous step by the SEWARD integral code, or alternatively (only for closed shell calculations) integral-direct by recomputing all the two-electron integrals when needed [124]. The later route is recommended for large basis sets or molecules, when the two-electron integral file would become extensively large. It is automatically taken, when the SCF program cannot find any *ORDINT* file in the work directory. The direct Fock matrix construction employs an efficient integral prescreening scheme, which is based on differential densities [125, 126]: only those AO integrals are computed, where the estimated contractions with the related differential density matrix elements give significant (Coulomb or exchange) contributions to the (differential) two-electron part of the Fock matrix. Integral prescreening is performed at two levels, (i) at the level of shell quadruples, and (ii) at the level of individual primitive Gaussians. Prescreening at the level of contracted functions is not supported, because this would be inefficient in the context of a general contraction scheme. In order to work with differential density and Fock matrices, a history of these entities over previous iterations has to be kept. All these matrices are partly kept in memory, and partly held on disk. The SCF program either works with simple differences of the actual and the previous density, or alternatively with minimized densities, obtained from linear combinations of the actual density and all the previous minimized densities.
Besides the conventional and the fully-direct algorithms there is also a semi-direct path, which allows for the storage of some of the AO integrals during the first iteration, which then are retrieved from disk in subsequent iterations. That path is taken, if the keyword DISK with an appropriate argument specifying the amount of AO integrals to store is found on the input stream. The semi-direct path is recommended for medium sized problems, where the two-electron integral file would become a bit too large (but not orders of magnitude).

The program contains a feature that allows you to make the orbitals partially populated during the aufbau procedure. This feature in not primarily intended to accelerate the convergence but rather to ensure that you do get convergence in difficult cases. The orbitals are populated with with electrons according to

$$\eta_i = \frac{2}{1 + e^{(\varepsilon_i - \varepsilon_f)/kT}} \tag{8.8}$$

where ε_i is the orbital energy of orbital i and ε_f is the Fermi energy. In this “Fermi aufbau” procedure the temperature is slowly lowered until it reaches a minimum value and then kept constant until a stable closed shell configuration is determined. Then normal SCF iterations will be performed with the selected closed shell configuration. For systems that are not really closed shell systems, for example diradicals, you might end up in the situation that the program does not find any stable closed shell configuration. In that case it will continue to optimize the closed shell energy functional with partial occupation numbers. If this is the case, this is probably what you want, and such orbitals would be ideal as starting orbitals for an MCSCF calculation.

The initial orbital guess is either obtained by diagonalizing the bare nuclei Hamiltonian, from an initial guess produced by the module `GUESSORB` or from orbitals of a previous Hartree-Fock SCF calculation. These starting orbitals are automatically located in the order

1. SCF orbitals from a previous calculation located in the *`RUNFILE`*
2. SCF orbitals from a previous calculation located in a formatted orbitals file, *`INPORB`*.
3. initial guess orbitals from module `GUESSORB` located in the *`RUNFILE`* and

The program has three types of convergence accelerating schemes: (i) dynamic damping [127], (ii) the C^2-DIIS method using the orbital gradient as error vector [128], and (iii) a combined second-order update/C^2-DIIS procedure. The latter eliminates the Brillouin violating elements of the Fock matrix by proper orbital rotations and hence avoids diagonalization of the Fock matrix: the approximate inverse Hessian is updated (BFGS) in a first step, and then the new orbital displacement vector is obtained from the updated Hessian using C^2-DIIS extrapolation [129]. Dynamic damping gives substantial improvements in highly anharmonic regions of the energy hyper surface, while the second-order update/C^2-DIIS procedure exhibits excellent convergence for less anharmonic regions. By default, dynamic damping is used during the first few iterations. When the change in the density between two subsequent iterations drops below a certain threshold the second-order update/C^2-DIIS procedure kicks in. It is also possible to use the older first order C^2-DIIS scheme instead of the second-order update/C^2-DIIS procedure by setting the density threshold for the latter to zero in the corresponding input card (keyword QNRTHRESHOLD).
It is possible to restart the SCF iteration (not, however, while Aufbau is in effect) after each cycle without loosing any information about the Hessian, which was accumulated in previous cycles. In fact, since for the first iteration after a restart the total density rather than a differential density is contracted with the integrals, it is recommended to restart after 5–15 iterations and gradually to decrease the SCF convergence thresholds. This increases both the accuracy of the final result, and the efficiency of prescreening in direct calculations.

By default `SCF` behaves in different ways depending on what kind of start orbitals are found according to

1. No start orbitals are found. In this case the core hamiltonian is diagonalized and these orbitals are used as start. The “Fermi aufbau” procedure is used until a stable configuration is found.
2. Start orbitals from `GUESSORB` are found. In this case the HOMO LUMO gap is analyzed and if it is small the “Fermi aufbau” procedure is used until a stable configuration is found. Otherwise the configuration suggested by `GUESSORB` is used.

3. Start orbitals from a previous `SCF` calculation is found. The configuration from the previous `SCF` calculation is used, unless some problem is detected such as partial occupation numbers from an unconverged calculation. In the latter case "Fermi aufbau" is used.
4. Start orbitals from an *`INPORB`* is in the same way as for start orbitals from an `SCF` calculation, see last point.

This behavior can be changed by suitable keywords described below.

One of the main objects of the `SCF` program in the context of the *MOLCAS* program system is to generate starting orbitals for subsequent MCSCF calculations. Two options are available to improve the canonical Hartree-Fock orbitals in this respect.
(i) It is possible to specify pseudo occupation numbers that are neither zero nor two, thus simulating to some extent an open shell system. The resulting wavefunction does not have any physical meaning, but will provide better starting orbitals for open shell systems.
(ii) Usually, the lowest virtual canonical Hartree-Fock orbitals are too diffuse as correlating orbitals in an MCSCF calculation. If the keyword IVO is encountered in the input stream, the `SCF` program will diagonalize the core Hamiltonian matrix within the virtual space and write the resulting more compact eigenvectors to the *`SCFORB`* and *`RUNFILE`* files, rather than the virtual eigenvectors of the Fock matrix. It should be noted, that this option must never be used, if the `SCF` wave function itself is used subsequently as a reference function: no MP2 or coupled cluster calculations after an `SCF` run with IVO !
A further method to generate starting orbitals for MCSCF calculations is to perform an SCF calculation for a slightly positively charged moiety.

8.32.2 Dependencies

The `SCF` program requires the one-electron integral file *`ONEINT`* and the communications file *`RUNFILE`*, which contains among others the basis set specifications processed by `SEWARD`. For conventional (not integral-direct) runs the two-electron integral file *`ORDINT`* is required as well. All these files are generated by a preceding `SEWARD` run.

8.32.3 Files

Below is a list of the files that are used/created by the program `SCF`.

Input files

File	*Contents*
`ONEINT`	One-electron integral file generated by the program `SEWARD`.
`RUNFILE`	File for communication of auxiliary information generated by the program `SEWARD`. Contains many things, e.g. the basis set specifications and is augmented with specific entries by most of the individual programs (also by `SCF`).

*ORDINT**	Ordered and packed two-electron integrals generated by the program `SEWARD` (possibly segmented into multiple files). Only used for conventional (not integral-direct) runs.
INPORB	*SCFORB* file containing the Hartree-Fock orbitals of a previous `SCF` run, which are used now as starting vectors in the present run.

Intermediate files

All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
DNSMAT,TWOHAM	History of differential density and two-electron Fock matrices. This file must be kept if a subsequent restart is desired.
GRADIENT	History of orbital gradients. This file must be kept if a subsequent restart is desired.
SO...	*SODGRAD*, *SOXVEC*, *SODELTA*, *SOYVEC*: History files for second-order update procedure. This file must be kept if a subsequent restart is desired.
TMPDIR	File containing the two-electron integrals that are stored in the 1st SCF iteration and retrieved in subsequent iterations, if the program follows the semi-direct path.

Output files

File	*Contents*
SCFORB	`SCF` orbital output file. Contains the canonical Hartree-Fock orbitals for closed shell calculations. If the IVO option was specified, the virtual orbitals instead are those that diagonalize the bare nuclei Hamiltonian within that subspace.
UHFORB	Contains the canonical Hartree-Fock orbitals for open shell calculations.
UNAORB	This file is produced if you make a UHF calculation and it contain natural orbitals.
RUNFILE	Communication file for subsequent programs.
MOLDEN	Molden input file for molecular orbital analysis.

8.32.4 Input

Below follows a description of the input to `SCF`. The keywords are always significant to four characters, but in order to make the input more transparent, it is recommended to use the full keywords. The input for each module is preceded by its name like:

```
␣&SCF
```

Argument(s) to a keyword are always supplied on the next line of the input file, except explicitly stated otherwise.

Basic general keywords

Below is a list of keywords that should cover the needs of most users.

Keyword	*Meaning*
TITLe	All lines following this line are regarded as title lines until the next keyword is encountered. Any number of title lines may be specified, but only the first 10 will be printed.
UHF	Use this keyword to run Unrestricted Hartree-Fock code. Note, that current implementation of UHF code has some restrictions, and not all features of SCF program are supported.
ZSPIN	Use this keyword to specify the difference in the number of α and β electron in the system. The default is 0 or 1 dependent on if there is an even or odd number of electrons. This keyword is not needed when you specify the number of electrons with the keyword OCCUpied.
KSDFT	Use this Keyword to do the Density Functional Theory Calculations. This Keyword should be followed in the next line by functional Keyword: BLYP, B3LYP, B3LYP5, HFB, HFS, LDA, LDA5, LSDA, LSDA5, SVWN, SVWN5, TLYP, PBE, PBE0, M06, M062X, M06HF, M06L.

CHARge	Use this keyword to set the number of electrons in the system. This number is defined by giving the net charge of the system. If this keyword is not specified, the molecule is assumed to have net charge zero. The input is given as
	`Charge`
	`␣n`
	where `n` is the charge of the system.

OCCUpied	Use this keyword to set the number of electrons in the system. This number is defined by giving the number of electron pairs per irreducible representation of the subgroup of D_{2h} used in the calculation. You can use one and only one of the keywords, CHARge and OCCUpied for this purpose. If neither of these keywords are specified

CHARge is assumed with a net charge of zero. It should be noted that the “fermi aufbau” procedure is not used when you specify this keyword. The input for one of the point groups D_2, C_{2h} or C_{2v} is given as

```
OCCUpied
␣n1␣n2␣n3␣n4
```

where `n1` is the number of electron pairs (occupied orbitals) in the first irreducible representation, etc.

If UHF keyword was specified, occupation numbers must be specified in two lines: for alpha and beta spins

FERMi

Use this keyword to specify that you want to use the “Fermi aufbau” procedure for the first few iterations to ensure convergence. The orbitals will be partially populated according to a Fermi population. The input is gives as

```
␣Fermi
␣␣m
```

where `m` is the temperature parameter according to

- `m=0`: No temperature is used. Not recommended.
- `m=1`: A low temperature is used and will yield swift convergence for well behaved systems.
- `m=2`: A medium low temperature is used and will yield swift and safe convergence for most systems. The is the default value.
- `m=3`: A medium temperature is used and you will obtain good convergence for closed shell systems. If the system is not a closed shell system, the temperature dependent aufbau procedure may not terminate. This will result in a density matrix with fractional occupation numbers.
- `m=4`: A medium high temperature is used and the temperature dependent aufbau procedure will most probably not terminate. This is useful for generating starting orbitals for an MCSCF calculation.
- `m=5`: A high temperature is used. Behaves as m=4 only more so.

It should be noted that only dynamic damping is used until the program have found a stable closed shell configuration. When this have happened the more efficient methods: the ordinary C2-DIIS and the second order update/C2-DIIS procedure, are enabled.

CHOLesky

`SCF` will use Cholesky (or RI/DF) representation of the two-electron integrals to compute the corresponding contributions to the Fock or KS matrices. The default (LK) algorithm is used. The configuration may be tailored using the ChoInput section. Default is to not use Cholesky unless the Cholesky (or RI/DF) representation of the two-electron integrals has been produced by `SEWARD`.

CHOInput	This marks the start of an input section for modifying the default settings of the Cholesky SCF. Below follows a description of the associated options. The options may be given in any order, and they are all optional except for **ENDChoinput** which marks the end of the **CHOInput** section. • **NoLK** - Available only within ChoInput. Deactivates the “Local Exchange” (LK) screening algorithm [121] in computing the Fock matrix. The loss of speed compared to the default algorithm can be substantial, especially for electron-rich systems. Default is to use LK. • **DMPK** - Available only within ChoInput. Modifies the thresholds used in the LK screening. The keyword takes as argument a (double precision) floating point (non-negative) number used as correction factor for the LK screening thresholds. The default value is 1.0d0. A smaller value results in a slower but more accurate calculation. **Note.:** The default choice of the LK screening thresholds is tailored to achieve as much as possible an accuracy of the converged SCF energy consistent with the choice of the Cholesky decomposition threshold. • **NODEcomposition** - Available only within ChoInput. Deactivates the Cholesky decomposition of the AO 1-particle density matrix. The Exchange contribution to the Fock matrix is therefore computed using occupied canonical orbitals instead of (localized) “Cholesky MOs” [111]. This choice tends to lower the performances of the LK screening. Default is to perform this decomposition in order to obtain the Cholesky MOs. • **TIME** - Activates printing of the timings of each task of the Fock matrix build. Default is to not show these timings. • **MEMFraction** - Set the fraction of memory to use as global Cholesky vector buffer. Default: for serial runs 0.0d0; for parallel runs 0.3d0.
ITERations	Specifies the maximum number of iterations. The default is 200 which is also the largest number you can specify.
CORE	The starting vectors are obtained from a diagonalization of the core Hamiltonian. No additional input is required.
LUMORB	The starting vectors are taken from a previous *`SCFORB`* file called *`INPORB`*. No additional input is required.
LUMORB	The starting vectors are taken from a previous *`SCFORB`* file, specified by user. No additional input is required.
GSSRunfile	The starting vectors are taken from the orbitals produced by `GUESSORB`. No additional input is required.

HLGAp	This keyword is used to make the program level shift the virtual orbitals in such a way that the HOMO LUMO gap is at least the value specified on the next line. This will help convergence in difficult cases but may lead to that it converges to an excited configuration. A suitable value is 0.2.

Advanced general keywords

Keyword	*Meaning*
SCRAmble	This keyword will make the start orbitals slightly scrambled, accomplished by making a few small random orbital rotations. How much the orbitals are scrambled is determined by the parameter read on the next line. A reasonable choise for this parameter is 0.2 which correspond to maximum rotation angle of $\arcsin 0.2$. Using this keyword may be useful for UHF calculations with same number of α and β electrons that are not closed shell cases.
ORBItals	Specifies the number of orbitals in the subspace of the full orbital space defined by the basis set, in which the SCF energy functional is optimized. The size of this subspace is given for each of the irreducible representations of the subgroup of D_{2h}. If this keyword is not specified when starting orbitals are read, the full orbital space is assumed. The keyword takes as argument *nIrrep* (# of irreps) integers. **Note** that this keyword is only meaningful when the `SCF` program is fed with input orbitals (cf. LUMORB).
FROZen	Specifies the number of orbitals not optimized during iterative procedure. The size of this subspace is given for each of the irreducible representations of the subgroup of D_{2h}. If this keyword is not specified the number of frozen orbitals is set to zero for each irreducible representation. If the starting vectors are obtained from a diagonalization of the bare nuclei Hamiltonian the atomic orbitals with the lowest one-electron energy are frozen. If molecular orbitals are read from *`INPORB`* the frozen orbitals are those that are read in first in each symmetry. The keyword takes as argument *nIrrep* (# of irreps) integers.
SPDElete	Specifies that the s-component of the 6 cartesian d-components is excluded from the calculation. Likewise the p-components of f-functions, and the s- and d-components of g-functions. No arguments are required.
OVLDelete	Specifies the threshold for deleting near linear dependence in the basis set. The eigenvectors of the overlap matrix with eigenvalues less than that threshold are removed from the orbital subspace, and do not participate in the optimization procedure. The default value is 1.0d-5. The keyword takes as argument a (double precision) floating

	point number. Note that the ***`SCFORB`*** file will contain the deleted orbitals as a complementary set to the actual SCF orbitals! In future use of this orbital file the complementary set should always be deleted from use.
PRORbitals	Specifies which orbitals are to be printed in the log file (standard output). The keyword takes as argument two integers. The possible values are: 0 — No orbitals printed. 1 — orbitals with orbital energies smaller than $2E_{\text{homo}} - E_{\text{lumo}}$ are printed. 2 — followed by real number (ThrEne) — orbitals with orbital energies smaller than ThrEne are printed. Default value is 1. Second (optional) argument specifies a format: 0 — No orbitals printed 1 — Print only one-electron energies and Occupation numbers 2 — Short print format 3 — Extended print format Default value is 3 for small numbers of MOs and 2 for number of MOs$>$ 256.
PRLScf	Specifies the general print level of the calculation. An integer has to be supplied as argument. The default value, 1, is recommended for production calculations.
RESTART	The SCF iteration is restarted without loosing any information obtained from previous cycles whatsoever. All the scratch files are required. No additional input has to be provided. Note that this option only works if the SCF job has not died and if Aufbau is not in effect. **Note.:** the keywords CORE, LUMORB, GSSRunfile and RESTART are mutually exclusive.
THREsholds	Specifies convergence thresholds. Four individual thresholds are specified as arguments, which have to be fulfilled simultaneously to reach convergence: EThr, DThr and FThr specify the maximum permissible difference in energy, density matrix elements and Fock matrix elements, respectively, in the last two iterations. The DltNTh finally specifies the norm of the orbital displacement vector used for the orbital rotations in the second-order/C^2-DIIS procedure. The corresponding values are read in the order given above. The default values are 1.0d-9, 1.0d-4, 1.5d-4, and 0.2d-4, respectively. **Note** that these thresholds automatically define the threshold used in the direct Fock matrix construction to estimate individual contributions to the Fock matrix such that the computed energy will have an accuracy that is better than the convergence threshold.

NODIis	Disable the DIIS convergence acceleration procedure. No additional input is required.
DIISthr	Set the threshold on the change in density, at which the DIIS procedure is turned on. The keyword takes as argument a (double precision) floating point number. The default value is 0.15.
QNRThr	Set the threshold on the change in density, at which the second-order/C^2-DIIS procedure kicks in. The keyword takes as argument a (double precision) floating point number. The default value is 0.15. **Note.:** the change in density has to drop under both the DIIS thr and the QNRThr threshold, for the second-order/C^2-DIIS to be activated. If the latter is set to zero the older first order C^2-DIIS procedure will be used instead.
C1DIis	Use C^1-DIIS for convergence acceleration rather that C^2-DIIS which is the default (not recommended). No additional input is required.
NODAmp	Disable the Damping convergence acceleration procedure. No additional input is required.
OCCNumbers	Gives the option to specify occupation numbers other than 0 and 2. This can be useful for generating starting orbitals for open shell cases. It should be noted however, that it is still the closed shell SCF energy functional that is optimized, thus yielding unphysical energies. Occupation numbers have to be provided for all occupied orbitals. In the case of UHF calculation occupation numbers should be specified on two lines: for alpha and beta spin.
IVO	Specifies that the virtual orbitals are to be improved for subsequent MCSCF calculations. The core Hamiltonian is diagonalized within the virtual orbital subspace, thus yielding as compact orbitals as possible with the constraint that they have to be orthogonal to the occupied orbitals. **Note** that this option must not be used whenever the Hartree-Fock wavefunction itself is used as a reference in a subsequent calculation. No additional input is required.
NOMInimization	Program will use density differences $D^{(\mathrm{k})} - D^{(\mathrm{k}-1)}$ rather than minimized differences. No additional input is required.
ONEGrid	Disable use of a smaller intermediate grid in the integration of the DFT functional during the first SCF iterations.
RFPErt	This keyword will add a constant reaction field perturbation to the bare nuclei hamiltonian. The perturbation is read from the file ***RUNFILE*** and is the latest self consistent perturbation generated by one of the programs `SCF` or `RASSCF`.
STAT	This keyword will add an addition print outs with statistic information.

For calculations of a molecule in a reaction field see section 8.14.1 of the present manual and section 5.6 of the examples manual.

DFT functionals: Below is listed the keywords for the DFT functionals currently implemented in the package.

Keyword	*Meaning*
LSDA, LDA, SVWN	Vosko, Wilk, and Nusair 1980 correlation functional fitting the RPA solution to the uniform electron gas [130] (functional III in the paper).
LSDA5, LDA5, SVWN5	Functional V from the VWN80 paper [130] which fits the Ceperley-Alder solution to the uniform electron gas.
HFB	Becke's 1988 exchange functional which includes the Slater exchange along with corrections involving the gradient of the density [131].
HFS	$\rho^{4/3}$ with the theoretical coefficient of 2/3 also known as Local Spin Density exchange [132, 133, 134].
BLYP	Becke's 1988 exchange functional which includes the Slater exchange along with corrections involving the gradient of the density [131]. Correlation functional of Lee, Yang, and Parr, which includes both local and non-local terms [135, 136].
B3LYP	Becke's 3 parameter functional [137] with the form $$A * E_X^{Slater} + (1-A) * E_X^{HF} + B * \Delta E_X^{Becke} + E_C^{VWN} + C * \Delta E_C^{non-local} \tag{8.9}$$ where the non-local correlation functional is the LYP functional and the VWN is functional III (not functional V). The constants A, B, C are those determined by Becke by fitting to the G1 molecule set, namely A=0.80, B=0.72, C=0.81.
B3LYP5	Becke's 3 parameter functional [137] with the form $$A * E_X^{Slater} + (1-A) * E_X^{HF} + B * \Delta E_X^{Becke} + E_C^{VWN} + C * \Delta E_C^{non-local} \tag{8.10}$$ where the non-local correlation functional is the LYP functional and the VWN is functional V. The constants A, B, C are those determined by Becke by fitting to the G1 molecule set, namely A=0.80, B=0.72, C=0.81.
TLYP	$$E_X^{HF} + E_C^{non-local} \tag{8.11}$$ where the non-local correlation functional is the LYP functional
PBE	The Perdew, Burke, Ernzerhof GGA functional 1996[138].
PBE0	The Perdew, Ernzerhof, Burke non-empirical hybrid functional 1996[139].
M06	The M06 functional of the Minnesota 2006 class of functionals by Zhao and Truhlar [140, 141, 142, 143]
M06-L	The M06-L functional of the Minnesota 2006 class of functionals by Zhao and Truhlar [140, 141, 142, 143]

M06-HF	The M06-HF functional of the Minnesota 2006 class of functionals by Zhao and Truhlar [140, 141, 142, 143]
M06-2X	The M06-2X functional of the Minnesota 2006 class of functionals by Zhao and Truhlar [140, 141, 142, 143]

Keywords for direct calculations

Note again that the threshold for contributions to the Fock matrix depends on the convergence thresholds mentioned above. The choice between the conventional and direct SCF methods is based on the presence of a two-electron integral file (file *`ORDINT`*). The keyword DIRECT in the `SEWARD` input controls that no two-electron integral file is to be generated and that integral direct algorithms can be used in subsequent modules. Thus, *the choice between conventional and direct SCF is done already in the input for the integral program* `SEWARD`. The direct (or semi-direct) path will be taken whenever there are no two-electron integrals available.

Keyword	*Meaning*
CONVENTIONAL	This option will override the automatic choice between the conventional and the direct SCF algorithm such that the conventional method will be executed regardless of the status of the *`ORDINT`* file.
DISK	This option enables/disables the semi-direct algorithm. It requires two arguments which specifies the max Mbyte of integrals that are written on disk during the first iteration (and retrieved later in subsequent iterations) and the size of the corresponding I/O buffer in kbyte. The default values are 2000 MByte and 512 kByte. In case the specified disk space is zero and the I/O buffer is different from zero it will default to a semi-direct SCF with in-core storage of the integrals. The size of the memory for integrals storage is the size of the I/O buffer. If the size of the disk is non-zero and the I/O buffer size is zero the latter will be reset to the default value.
THIZE	This option specifies a threshold for two-electron integrals. Only integrals above this threshold (but not necessarily all of those) are kept on disk for the semi-direct algorithm. The keyword takes as argument a (double precision) floating point number.
SIMPLE	If this option is specified, only a simple prescreening scheme, based solely on the estimated two-electron integral value will be employed (no density involved).

Limitations

The limitations/MODULE on the number of basis functions are the same as specified for `SEWARD`.

Input examples

First we have the bare minimum of input. This will work well for almost all systems containing an even number of electrons.

```
␣&SCF
End␣of␣Input
```

The next example is almost as simple. Here we have an open shell case, i.e. you have an odd number of electrons in the neutral system and you need to generate starting orbitals for RASSCF. In this case we recommend that you perform a calculation on the cation with the input below.

```
␣&SCF
Charge
␣1
End␣of␣input
```

The next example explains how to run UHF code for a nitrogen atom:

```
␣&SCF
UHF
ZSPIN
␣3
End␣of␣input
```

The next example is a bit more elaborate and show how to use a few of the keywords. The system is water that have the electron configuration $1a_1^2 2a_1^2 3a_1^2 1b_1^2 1b_2^2$.

```
␣&SCF
Title
␣Water␣molecule.
␣Experimental␣equilibrium␣geometry.
*␣The␣symmetries␣are␣a1,␣b2,␣b1␣and␣a2.
Occupied
␣3␣1␣1␣0
*␣15␣RHF␣iterations
Iterations
␣15
*␣convergence␣threshold/MODULEs
*␣EThr␣DThr␣FThr␣DltNTh
Threshold
␣0.5D-9␣0.5D-6␣0.5D-6␣0.5D-5
*␣semi-direct␣algorithm␣writing␣max␣128k␣words␣(1MByte)␣to␣disk
*␣the␣size␣of␣the␣I/O␣buffer␣by␣default␣(512␣kByte)
Disk
␣1␣0
*␣Improve␣the␣virtuals␣for␣MCSCF.
Ivo
End␣of␣input
```

8.33 SEWARD

The Seward module generates one- and two-electron integrals needed by other programs. The two-electron integrals may optionally be Cholesky decomposed. In addition, it will serve as the input parser for parameters related to the specification of the quadrature grid used in numerical integration in association with DFT and reaction field calculations.

Figure 8.2: H. W. Seward, secretary of State in the Lincoln administration, who suggested and supervised the 1867 purchase of Alaska from tzar Russia. Price: 2 cents an acre.

In the following three subsection we will in detail describe the input parameters for analytic integration, numerical integration, and reaction fields.

8.33.1 Analytic integration

Any conventional ab initio quantum chemical calculation starts by computing overlap, kinetic energy, nuclear attraction and electron repulsion integrals. These are used repeatedly to determine the optimal wave function and the total energy of the system under investigation. Finally, to compute various properties of the system additional integrals may be needed, examples include multipole moments and field gradients.

Description

`SEWARD` is able to compute the following integrals:

- kinetic energy,
- nuclear attraction,
- two electron repulsion (optionally Cholesky decomposed),
- n'th (default n=2) order moments (overlap, dipole moment, etc.),
- electric field (generated at a given point by all charges in the system),
- electric field gradients (spherical gradient operators),
- linear momentum (velocity),
- orbital angular momentum,
- relativistic mass-velocity correction (1st order),
- one-electron Darwin contact term,
- one-electron relativistic no-pair Douglas-Kroll
- diamagnetic shielding,
- spherical well potential (Pauli repulsion),
- ECP and PP integrals,
- modified kinetic energy and multipole moment integrals (integration on a finite sphere centered at the origin) for use in the variational R-matrix approach,
- external field (represented by a large number of charges and dipoles),

- angular momentum products, and
- atomic mean-field integrals (AMFI) for spin-orbit coupling.

In general these integrals will be written to a file, possibly in the form of Cholesky vectors (two-electron integrals only). However, `SEWARD` can also compute the orbital contributions and total components of these properties if provided with orbital coefficients and orbital occupation numbers.

To generate the one- and two-electron integrals `SEWARD` uses two different integration schemes. Repulsion type integrals (two- electron integrals, electric field integrals, etc.) are evaluated by the reduced multiplication scheme of the Rys quadrature [144]. All other integrals are computed by the Gauss-Hermite quadrature. `SEWARD` use spherical Gaussians as basis functions, the only exeption to this are the diffuse/polarization functions of the 6-31G family of basis sets. The double coset [145] formalism is used to treat symmetry. `SEWARD` is especially designed to handle ANO-type basis sets, however, segmented basis sets are also processed.

At present the following limitations are built into `SEWARD`:

Max number of unique basis functions:	2000
Max number of symmetry independent centers:	500
Highest angular momentum:	15
Highest symmetry point group:	D_{2h}

Dependencies

`SEWARD` does normally not depend on any other code. There are two exceptions however. The first one is when `SEWARD` is used as a property module. In these cases the file *`INPORB`* has to have been generated by a wave function code. The second case, which is totally transparent to the user, is when `SEWARD` picks up the new Cartesian coordinates generated by `SLAPAF` during a geometry optimization.

Files

Input Files Apart form the standard input file `SEWARD` will use the following input files.

File	*Contents*
`RYSRW`	Data base for the fast evaluation of roots and weights of the Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
`ABDATA`	Data base for the evaluation of roots and weights of high order Rys polynomial. This file is a part of the program system and should not be manipulated by the user.
`BASLIB`	The default directory for one-particle basis set information. This directory contains files which are part of the program system and could be manipulated by the user in accordance with the instructions in the section 8.36 and following subsections. New basis set files can be added to this directory by the local *MOLCAS* administrator.

QRPLIB	Library for numerical mass-velocity plus Darwin potentials (used for ECPs).
INPORB	Orbital coefficients and occupation numbers of natural orbitals.
RUNFILE	File for communication of auxiliary information.

Intermediate files All the intermediate files are created, used and removed automatically, unless you yourself create a link or a file with the specified name.

File	*Contents*
*TEMP01**	scratch file for integral packing (approximate size, 1 Byte/two-electron integral).

Output files In addition to the standard output file SEWARD may generate the following files.

File	*Contents*
ONEINT	One-electron integrals and auxiliary information.
BASINT	Conventional two-electron integral file. For compatibility with *MOLCAS*-1.
*ORDINT**	Ordered and packed two-electron integral file (not generated if the Cholesky decomposition option is turned on).
*CHVEC**	Files for storing Cholesky vectors (separate files for each irreducible representation, the base file name being CHVEC1 for irreducible representation 1, CHVEC2 for irreducible representation 2, and so forth. Each file CHVEC1, CHVEC2, etc. may be splitted, hence the asterisk in the base name above).
*CHRED**	File for storing index arrays for addressing Cholesky vectors.
CHODIAG	File for reduced integral diagonal (needed for diagonal restart calculations).
*CHORST**	File containing information about Cholesky decomposition procedure (needed for decomposition restart calculations).
CHOMAP	Integral calculation map for Cholesky decomposition (needed for decomposition restart).
CHOR2F	File containing the mapping between Cholesky vector storage and the canonical orbital ordering. Needed by all modules using the Cholesky integral representation.
RUNFILE	File for communication of auxiliary information.

One-Electron Integral Labels

The storage of one-electron integrals on disk is facilitated by the one-electron integral I/O facility. The internal structure of the one-electron file and the management is something which the user normally do not need to worry about. However, for the general input section of the `FFPT`, the user need to know the name and structure of the internal labels which the one-electron integral I/O facility associates with each type of one-electron integral. The labels are listed and explained here below for reference. The component index is also utilized by the one-electron integral I/O facility to discriminate the various components of the one-electron integrals of a certain type, for example, the dipole moment integrals have three components (1=x-component, 2=y-component, 3=z-component). The component index is enumerated as a canonical index over the powers of the Cartesian components of the operator (e.g. multipole moment, velocity, electric field, etc.). The order is defined by following pseudo code,

```
Do ix = nOrder, 0, -1
   Do iy = nOrder-ix, 0, -1
      iz = nOrder-ix-iy
   End Do
End Do,
```

where nOrder is the total order of the operator, for example, nOrder=2 for the electric field gradient and the quadrupole moment operator.

Label	Explanation
'Mltpl *nn*'	the *nn*'th order Cartesian multipole moments.
'MltplS '	the overlap matrix used in the semi-empirical NDDO method.
'Kinetic '	the kinetic energy integrals.
'Attract '	the electron attraction integrals.
'AttractS'	the electron attraction integrals used in the semi-empirical NDDO method.
'PrjInt '	the projection integrals used in ECP calculations.
'M1Int '	the M1 integrals used in ECP calculations.
'M2Int '	the M2 integrals used in ECP calculations.
'SROInt '	the spectrally resolved operator integrals used in ECP calculations.
'XFdInt '	the external electric field integrals.
'MassVel '	the mass-velocity integrals.
'Darwin '	the Darwin one-electron contact integrals.
'Velocity'	the velocity integrals.
'EF0nnnnn'	the electric potential at center *nnnnn*.
'EF1nnnnn'	the electric field at center *nnnnn*.
'EF2nnnnn'	the electric field gradient at center *nnnnn*.
'AngMom '	the angular momentum integrals.
'DMS '	the diamagnetic shielding integrals.
'Wellnnnn'	the *nnnn*'th set of spherical well integrals.
'OneHam '	the one-electron Hamiltonian.
'AMProd '	the hermitized product of angular momentum integrals.
'AMFI '	the atomic mean field integrals.

Input

Below follows a description of the input to `SEWARD`. Note that `SEWARD` as a minimum input requires a basis set definition (keyword BASIS SET) with at least one center. All other input, apart from the END OF INPUT card, is optional. Observe that if nothing else is requested `SEWARD` will by default compute the overlap, the dipole, the quadrupole, the nuclear attraction, the kinetic energy, the one-electron Hamiltonian, and the two-electron repulsion integrals.

In addition to the keywords and comment lines the input may contain blank lines. The keywords are always significant to four characters. To make the input more transparent we recommend the user to use the full keyword. The input is always preceded by the program reference

```
␣&SEWARD
```

General keywords

Keyword	*Meaning*
TITLE	Title card follows on one to ten lines. The default is no title cards.
TEST	`SEWARD` will only process the input and generate a non-zero return code.
ONEONLY	`SEWARD` will not compute the two-electron integrals.
NODKROLL	`SEWARD` will not compute Douglas-Kroll integrals.
DIRECT	Prepares for later integral-direct calculations. As with keyword ONEONLY, `SEWARD` will evaluate no two-electron integrals.
EXPERT	Sets "expert mode", in which various default settings are altered. Integral-direct calculations will be carried out if the two-electron integral file is unavailable.
CHOLESKY	`SEWARD` will Cholesky decompose the two-electron integrals using default configuration (in particular, the decomposition threshold is 1.0d-4) of the decomposition driver. The configuration may be tailored using the ChoInput section. Default is to not decompose.
LOW CHOLESKY	`SEWARD` will Cholesky decompose the two-electron integrals using low-accuracy (threshold 1.0d-4) configuration of the decomposition driver (the configuration may be tailored using the ChoInput section). Default is to not decompose.
MEDIUM CHOLESKY	`SEWARD` will Cholesky decompose the two-electron integrals using medium-accuracy (threshold 1.0d-6) configuration of the decomposition driver (the configuration may be tailored using the ChoInput section). Same as CHOLESKY. Default is to not decompose.

HIGH CHOLESKY	`SEWARD` will Cholesky decompose the two-electron integrals using high-accuracy (threshold 1.0d-8) configuration of the decomposition driver (the configuration may be tailored using the ChoInput section). Default is to not decompose.
JMAX	The integer entry on the next line is the highest rotational quantum number for which `SEWARD` will compute the rotational energy within the rigid rotor model. The default value is 5.
SYMMETRY	See the the description in the manual for the program `GATEWAY`
BASIS SET	See the the description in the manual for the program `GATEWAY`
ZMAT	See the the description in the manual for the program `GATEWAY`
NOGUESSORB	Disable automatic generation of starting orbitals with the GuessOrb procedure.
NODELETE	Do not delete any orbitals automatically.
SDELETE	Set the threshold for deleting orbitals based on the eigenvalues of the overlap matrix. All eigenvalues with eigenvectors below this threshold will be deleted. If you want no orbitals deleted use keyword NODELETE.
TDELETE	Set the threshold for deleting orbitals based on the eigenvalues of the kinetic energy matrix. All eigenvalues with eigenvectors above this threshold will be deleted. If you want no orbitals deleted use keyword NODELETE.
ECPSHOW	Force Seward to print ECP parameters.
AUXSHOW	Force Seward to print auxiliary basis set parameters.
BSSHOW	Force Seward to print basis set parameters.
VERBOSE	Force Seward to print a bit more verbose.

Cholesky specific keywords

Keyword	*Meaning*
CHOINPUT	This marks the start of a Cholesky input section for modifying the default configuration of the decomposition driver. Below follows a description of the options associated with the decomposition configuration. The options may be given in any order, and they are all optional except for **ENDChoinput** which marks the end of the **CHOInput** section. Keywords affecting the accuracy of the decomposition:

- **THRCholesky** - Set threshold for convergence of the Cholesky decomposition. The real entry on the subsequent line specifies the threshold. This keyword defines the formal accuracy of the decomposition. However, it may also be affected by the screening settings specified by the following keywords. Default value: 1.0D-4.
- **PREScreen** - Turn on diagonal prescreening. Default is off.
- **PREThreshold** - Turn on diagonal prescreening and set threshold. The real entry on the subsequent line specifies the diagonal prescreening threshold. Default is 1.0d-16.
- **NOPRescreen** - Turn off diagonal prescreening. This is the default.
- **DMP1** - Set "first" screening damping for setting up the first reduced set (i.e., the effective dimension of the diagonal and, consequently, of the Cholesky vectors). The real entry on the subsequent line specifies the damping. Default value ranges from 1.0d5 to 1.0d0 depending on the decomposition threshold.
- **DMP2** - Set "second" screening damping for setting up the second (and later) reduced set (i.e., the effective dimension of the updated diagonal and, consequently, of the Cholesky vectors). The real entry on the subsequent line specifies the damping. Default value ranges from 1.0d5 to 1.0d0 depending on the decomposition threshold.
- **SCREen** - Enable screening of diagonal during decomposition (using the first and second screening dampings given above). Default is to use screening (making this keyword redundant).
- **NOSCreen** - Disable screening of diagonal during decomposition (in which case the dampings above are only used for statistics). Default is to use screening.
- **ABSOlute** - Use absolute value of diagonal elements in screening procedure. This may leave in small negative diagonal elements arising from numerical inprecision. The default is to use absolute value.
- **NOABsolute** - Do not use absolute value of diagonal elements in screening procedure. This will eliminate any small negative diagonal elements arising from numerical inprecision. The default is to use absolute value.

Special keyword needed to run the initial Cholesky version of CASPT2:

- **REORder vectors** - Reorder vectors into full (i.e., canonical) storage after decomposition. The resulting files are significantly larger than the original ones.

Keywords for selecting decomposition algorithm:

- **TWOStep** - Use two-step algorithm. This is the default algorithm, as it is usually the fastest. However, the performance is dependent on memory availability. NOTE: the two-step algorithm can not be restarted!

- **ONEStep** - Use one-step algorithm. This is the original algorithm, not used by default.
- **NAIVe** - Use "naive" algorithm. Stop after the first step of the two-step algorithm. Not recommended, as it is highly inaccurate.
- **PARAllel** - Emulate the special algorithm used for parallel Cholesky decomposition. Mostly for debugging purposes.

Keywords for controlling output:

- **PRINt** - Set print level. The integer entry on the subsequent line specifies the print level. Default value: 1. (translation of print levels: 0 – silent, 1 – terse, 2 – terse but with additional timings info, 3 – detailed, 4 – verbose, 5 – debug, 6 or larger – insane.)

Keywords for restarting the decomposition (note that restart is not possible in parallel):

- **RSTDiagonal** - Do not compute initial diagonal. Instead, read diagonal from file CHODIAG which must be available. Also needed is the CHRED* file(s). Default is to not restart.
- **RSTCholesky** - Restart Cholesky decomposition procedure from Cholesky vectors available on disk using default restart model (see below). Files needed: CHVEC*, CHRED*, CHORST*, and CHOMAP. Note that currently it is not possible to restart calculations using the **TWOStep** algorithm. Default is to not restart.
- **RSTModel** - Set Cholesky decomposition restart model. The integer entry on the subsequent line specifies the restart model (-1 = use configuration from restart file and ignore input configuration, 0 = abort if discrepancies are detected between input and configuration on disk, +1 = use input configuration and ignore configuration on disk). Default value: -1. Note: setting the restart model will automatically trigger a restart (i.e., **RSTCholesky** is a short-hand for the default restart model). Also, note that currently it is not possible to restart calculations using the **TWOStep** algorithm.

Keywords specifying dimensions of some central index arrays:

- **CHOMax** - Set the maximum number of Cholesky vectors per irreducible representation. The integer entry on the subsequent line specifies the maximum. This is used to allocate index arrays. Default value: 20 times the number of basis functions in the largest irreducible representation.

- **REDMax** - Set the maximum number of reduced sets (i.e., integral passes). The integer entry on the subsequent line specifies the maximum. This is used to allocate index arrays. Default value: the number of irreducible representations times the maximum number of Cholesky vectors.

Keywords for setting algorithmic details:

- **VBUFfer** - Set the fraction of memory to use as global Cholesky vector buffer. Default: 0.35d0.
- **SPAN** - Set the span defining the max. ratio between the qualified and globally largest diagonals allowed during decomposition. The real entry on the subsequent line specifies the span. Default value: 1.0D-2.
- **MINQualified** - Set the minimum number of qualified diagonals needed to proceed to decomposition procedure. The integer value on the subsequent line specifies the minimum. Default value: 50.
- **MAXQualified** - Set the maximum number of diagonals that can be qualified per irreducible representation. The integer entry on the subsequent line specifies the maximum. Default value: 100.
- **QFRAction** - Set the memory fraction that may be used to store qualified integral columns. during integral evaluation as well as during vector computation. Thus, this may force the number of qualified to be less than the minimum number given through **MINQualified** (thereby saving memory for other processes such as the reading of previous vectors). The two integers (N1 N2) on the subsequent line specify the ratio N1/N2 of available memory. Default values: N1=1, N2=3.
- **MXSHell pair** - Set the maximum number of shell pair distributions (**—AB) that are allowed to be calculated before proceeding to decomposition procedure. The integer value on subsequent line specifies the maximum. Default is generic: calculate as many shell pair distributions as needed to meet the **MINQualified** requirement above.
- **ADDRessing** - Set type of I/O used for Cholesky vectors (1 for word-addressable files, 2 for direct-access files). The integer on the next line specifies the addressing mode. Default: 1 (WA-files).
- **IOVEctor** - Set algorithm used for reading vectors. The integer on the subsequent line specifies the algorithm according to
    1. Read vectors from same reduced set (rs) and copy from rs to rs (rs2rs/batch algorithm).
    2. Read vectors across reduced sets into a small buffer and do rs2rs copying (buffer/rs2rs algorithm). The size of the buffer is determined on the basis of memory availability.

  3. Read vectors across reduced sets into a large buffer without reordering at the time of read. (lrgbuf/rs2rs). The size of the buffer is determined on the basis of memory availability and as close as possible to the memory fraction specified by keyword **FRACtion**.
  4. Read vectors across reduced sets into a fixed-size buffer without reordering at the time of read. (fxdbuf/rs2rs). The size of the buffer can be controlled by the keyword **FRACtion**.

  The default is 3.

- **FRACtion** - Set the memory fraction that may be used as I/O vector buffer. The two integers (N1 N2) on the subsequent line specify the ratio N1/N2 of available memory. Default values: N1=2, N2=3. Note that the this is the fraction of memory available at the time of reading the vectors from disk and that the buffer is de-allocated again after reading. Thus, the N1/N2 fraction of memory will not interfere with the memory availability in other parts of the code such as integral evaluation.
- **MXSUbtraction** - Set the max. number of vectors in the subtraction part (i.e. dimension the matrix multiplication). The integer on the subsequent line specifies this number. Default: max. number of qualified columns (set by keyword **MAXQualified**.
- **IFCSeward** - Set interface to the integral evaluation of Seward (1 for storing full integral shell quadruple, 2 for extracting integrals directly in reduced set). Default: 2 (extract in reduced set). Note that 1 (store full shell quadruple) requires significantly more memory.

Keywords for calculating the integral diagonal:

- **BUFFersize** - Set size (in double precision words) of the buffer used during evaluation of the integral diagonal. The integer entry on the subsequent line specifies the size. Default value: 1000000 (or the amount needed to store the entire diagonal, if this is smaller).
- **THRDiagonal** - Set screening threshold for initial diagonal. The real entry on the subsequent line specifies the threshold. Default value: 0.0D0. WARNING: all integral diagonals smaller than the threshold will be unconditionally discarded in the course of calculating the initial diagonal.

Keywords for debugging:

- **CHECk configuration only** - Abort calculation after checking decomposition configuration. May be used to check input for inconsistencies. Default is to not abort.
- **CHKAll integrals** - Check all integrals after completing decomposition. This option is mostly for debugging and consumes significant amounts of CPU time. Default is to not check.

- **CHKSpecified integral columns** - Check specified integral columns (shell pairs) after completing decomposition. The integer entry on the subsequent line specifies the number of shell pair columns to check. This option is mostly for debugging and consumes significant amounts of CPU time. Default is to not check.
- **CHKMinimum number of integral columns** - Check a minimal integral columns (shell pairs) after completing decomposition. Which columns are checked depends on the decomposition at hand. This option is mostly for debugging and may consume significant amounts of CPU time. Default is to not check.
- **DIACheck** - Check the integral diagonal during decomposition by computing the diagonal from Cholesky vectors and comparing to the one stored in core. The double precision number on the subsequent line specifies the tolerance of the check. Default is not to check.
- **HALT** - Halt execution after decomposition. Default is not to halt.
- **TRCNegative** - Trace negative diagonal elements during decomposition. Default is not to trace.

Finally,

- **ENDChoinput** - Marks the end of the Cholesky input section. This card is mandatory.

Keywords associated to one-electron integrals

Keyword	*Meaning*
MULTipoles	Followed by a card which specifies the highest order of the multipole for which integrals will be generated. The default center for the dipole moment operator is the origin. The default center for the higher order operators is the center of the nuclear mass. The default is to do up to quadrupole moment integrals (2).
CENTer	This option is used to override the default selection of the origin of the multipole moment operators. This keyword followed by a card with an integer entry specifying the number of multipole moment operators for which the origin of expansion will be defined. Following this, one card for each operator, the order of the multipole operator and the coordinates of the center (in a.u.) of expansion are specified.
SDIPole	Supplement *`ONEINT`* for transition dipole moment calculations, i.e. dipole moment and velocity integrals will be computed. This option should be used whenever the `RASSI` program is used to compute transition moments, so that the transition moments can be evaluated in both velocity and length representation.

ANGM	Supplement ***ONEINT*** for transition angular momentum calculations. The keyword is followed by a card which specifies the angular momentum origin (in a.u.).
DSHD	Requests the computation of diamagnetic shielding integrals. The first subsequent card specifies the gauge origin. Then follows a card with an integer specifying the number of points at which the diamagnetic shielding will be computed. If this entry is zero, the diamagnetic shielding will be computed at each nucleus. If nonzero, then the coordinates (in a.u.) for each origin has to be supplied, one card for each origin.
RELInt	Requests the computation of mass-velocity and one-electron Darwin contact term integrals for the calculation of a first order correction of the energy with respect to relativistic effects.
AMPR	Request the computation of angular momentum product integrals. The keyword is followed by a card which specifies the angular momentum origin (in a.u.).
RXXPyy	Request arbitrary scalar relativistic Douglas-Kroll-Hess (DKH) correction to the one-electron Hamiltonian and the so-called picturer-change correction to the property integrals (multipole moments and eletronic potential related properties). Here XX represents the order of the DKH correction to the one-electron Hamiltonian and yy the order of the pciture-change correction. The character P denotes the parameterization used DKH procedure. The possible parametrizations P of the unitary transformation used in the DKH transformation supported by *MOLCAS*are: `P=O:` Optimum parametrization (OPT) `P=E:` Exponential parametrization (EXP) `P=S:` Square-root parametrization (SQR) `P=M:` McWeeny parametrization (MCW) `P=C:` Cayley parametrization (CAY) Hence, the proper keyword for 4th order relativistically corrected one-electron Hamiltonian and 3rd order relativistically corrected property integrals in the EXP parameterization would read as R04E03. If yy is larger than XX it is set to XX. If yy is omitted it will default to 2nd order. Recommended orders and parametrization is R02O.
NOAMfi	Explicit request for no computation of atomic mean-field integrals.
AMFI	Explicit request for the computation of atomic mean-field integrals (used in subsequent spin-orbit calculations). These integrals are computed by default for the ANO-RCC and ANO-DK3 basis sets.
EPOT	Followed by a card with an integer entry which represents the number of points for which the electric potential will be computed. If this number is zero, the electric field acting on each nucleus will be

	computed. If nonzero, then the coordinates (in a.u) for each point have to be supplied, one line for each point.
EFLD	Followed by a card with an integer entry which represents the number of points for which the electric potential and electric field will be computed. If this number is zero, the electric field acting on each nucleus will be computed. If nonzero, then the coordinates (in a.u) for each point have to be supplied, one line for each point.
FLDG	Followed by a card with an integer entry which represents the number of points for which the electric potential, electric field and electric field gradient will be computed. If this number is zero, the electric field gradient acting on each nucleus will be computed. If nonzero, then the either coordinates (in a.u.) for each point or labels for each atom center have to be supplied, one line for each point. In case a label i supplied it must match one of those given previous in the input during specification of the coordinates of the atom centers. Using a label instead of a coordinate can e.g. be useful in something like a geometry optimization where the coordinate isn't known when the input is written.
Grid Input	Specification of numerical quadrature parameters, consult the numerical quadrature section of this manual.

Additional keywords for property calculations

Keyword	*Meaning*
VECTors	Requests a property calculation. For this purpose a file, ***INPORB***, must be available, which contains the MO's and occupation numbers of a wave function.
ORBCon	The keyword will force `SEWARD` to produce a list of the orbital contributions to the properties being computed. The default is to generate a compact list.
THRS	The real entry on the following line specifies the threshold for the occupation number of an orbital in order for the OrbCon option to list the contribution of that orbital to a property. The default is 1.0d-6.

Keywords for two-electron integrals

Keyword	*Meaning*
NOPAck	The two-electron integrals will not be packed. The default is to pack the two-electron integrals.

PKTHRE	The next line specifies the desired accuracy for the packing algorithm, the default is 1.0d-10.
STDOUT	Generate a two-electron integral file according to the standard of version 1 of MOLCAS. The default is to generate the two-electron integrals according to the standard used since version 2 of MOLCAS.
THRESHOLD	Threshold for writing integrals to disk follows on next line. The default is 1.0d-10.
CUTOFF	Threshold for ignoring the calculation of integrals based on the pair prefactor follows on the next line. The default is 1.0d-10.

Keywords associated to electron-molecule scattering calculations within the framework of the R-matrix method This section contains keyword which control the radial numerical integration of the diffuse basis functions describing the scattered electrons in the variational R-matrix approach. The activation of this option is controlled by that the center of the diffuse basis is assigned the unique atom label DBAS.

Keyword	*Meaning*
RMAT	Radius of the R-matrix sphere (in Bohr). This sphere is centered at the coordinate origin. The default is 10 Bohr.
RMEA	Absolute precision in radial integration. The default is 1d-9.
RMER	Relative precision in radial integration. The default is 1d-14.
RMQC	Effective charge of the target molecule. This is the effective charge seen by the incident electron outside of the R-matrix sphere. The default is 0d0.
RMDI	Effective dipole of the target molecule. This is the effective dipole seen by the incident electron outside of the R-matrix sphere. The default is (0d0,0d0,0d0).
RMEQ	Minimal value of the effective charge of the target molecule to be considered. This is also the minimal value of the components of the effective dipole to be considered. Default is 1d-8
RMBP	Parameter used for test purposes in the definition of the Bloch term. Default is 0d0.
CELL	Defines the three vectors of the unit cell ($\vec{e_1}$,$\vec{e_2}$,$\vec{e_3}$). The optional keyword *Angstrom* before the definition of vectors would read data in $\AA$. Must consist of three lines (four in the case of $\AA$) which correspond to coordinates of the vectors. All the atoms which are defined after that key are considered as the atoms of the cell.

SPREAD	Three integer numbers n_1, n_2, n_3 which define the spread of the unit cell along the unit cell vectors. For example, 0 0 2 would add all cell's atoms translated on $-2\vec{e_3}$, $-\vec{e_3}$, $\vec{e_3}$, $2\vec{e_3}$. This key must be placed **before** the definition of the unit cell atoms.

Below follows an input for the calculation of integrals of a carbon atom. The comments in the input gives a brief explanation of the subsequent keywords.

```
␣&SEWARD
*␣Remove␣integrals␣from␣a␣specific␣irreps
Skip
0␣0␣0␣0␣1␣1␣1␣1
*␣Requesting␣only␣overlap␣integrals.
Multipole
0
*␣Request␣integrals␣for␣diamagnetic␣shielding
DSHD
0.0␣0.0␣0.0
1
0.0␣0.0␣0.0
*␣Specify␣a␣title␣card
Title
This␣is␣a␣test␣deck!
*␣Request␣only␣one-electron␣integrals␣to␣be␣computed
OneOnly
*␣Specify␣group␣generators
Symmetry
X␣Y␣Z
*␣Specify␣basis␣sets
Basis␣set
C.ANO-L...6s5p3d2f.
Contaminant␣d
C␣␣0.0␣0.0␣0.0
End␣of␣basis
```

The basis set label and the all electron basis set library The label, which defines the basis set for a given atom or set of atoms, is given as input in the row following the keyword Basis set. It has the following general structure (notice that the last character is a period):

atom.type.author.primitive.contracted.aux.

where the different identifiers have the following meaning:

Identifier	*Meaning*
`atom`	Specification of the atom by its chemical symbol.
`type`	Gives the type of basis set (ANO, STO, ECP, etc.) according to specifications given in the basis set library, *vide supra.* Observe that the upper cased character of the type label defines the file name in the basis directory.
`author`	First author in the publication where that basis set appeared.
`primitive`	Specification of the primitive set (e.g. 14s9p4d3f).

`contracted`	Specification of the contracted set to be selected. Some basis sets allow only one type of contraction, others all types up to a maximum. The first basis functions for each angular momentum is then used. **Note**, for the basis set types ANO and ECP, on-the-fly decontraction of the most diffuse functions are performed in case the number of contracted functions specified in this field exceeds what formally is specified in the library file.
`aux`	Specification of the type of AIMP, for instance, to choose between non-relativistic and relativistic core AIMP's.

Only the identifiers `atom`, `type`, and `contracted` have to be included in the label. The others can be left out. However, the periods have to be kept. Example — the basis set label '`C.ano-s...4s3p2d.`' will by *MOLCAS* be interpreted as '`C.ano-s.Pierloot.10s6p3d.4s3p2d.`', which is the first basis set in the ANO-S file in the basis directory that fulfills the specifications given.

Basis set format The **Inline** option for a basis set will read the basis set as defined by the following pseudo code.

```
      Read Charge, lAng
      Do 10 iAng = 0, lAng
         Read nPrim, nContr
         Read (Exp(iPrim),iPrim=1,nPrim)
         Do 20 iPrim=1,nPrim
            Read (Coeff(iPrim,iContr),iContr=1,nContr)
 20      Continue
 10   Continue
```

where *Charge* is the nuclear charge, *lAng* is the highest angular momentum quantum number, *nPrim* is the number of primitive functions (exponents) for a given shell, and *nContr* is the number of contracted functions for a given shell.

The following is an example of a DPZ basis set for carbon.

```
Basis␣set␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣Start␣defining␣a␣basis␣set
C.....␣/␣inline␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣Definition␣in␣input␣stream
␣␣␣␣␣␣␣6.0␣␣␣␣␣␣␣␣␣␣␣␣␣␣2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣charge,␣max␣l-quantum␣no.
␣␣␣␣9␣␣␣␣4␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣no.␣of␣prim.␣and␣contr.␣s-functions
4232.61␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣s-exponents
634.882
146.097
42.4974
14.1892
1.9666
5.1477
0.4962
0.1533
␣␣.002029␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣␣␣--␣s-contraction␣matrix
␣␣.015535␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.075411␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.257121␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.596555␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.242517␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.0␣␣␣␣␣␣␣1.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0
␣␣.0␣␣␣␣␣␣␣␣.0␣␣␣␣␣␣1.0␣␣␣␣␣␣␣.0
␣␣.0␣␣␣␣␣␣␣␣.0␣␣␣␣␣␣␣.0␣␣␣␣␣␣1.0
␣␣␣␣5␣␣␣␣2␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣no.␣of␣prim.␣and␣contr.␣p-functions
```

```
18.1557␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣p-exponents
3.98640
1.14290
0.3594
0.1146
␣.018534␣␣␣.0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣p-contraction␣matrix
␣.115442␣␣␣.0
␣.386206␣␣␣.0
␣.640089␣␣␣.0
␣.0␣␣␣␣␣␣␣1.0
␣␣␣␣1␣␣␣␣1␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣no.␣of␣prim.␣and␣contr.␣d-functions
␣␣␣.75␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣d-exponents
␣␣1.0␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣d-contraction␣matrix
C1␣0.00000␣0.00000␣0.00000␣␣␣␣␣␣␣␣␣␣␣␣␣--␣atom-label,␣Cartesian␣coordinates
C2␣1.00000␣0.00000␣0.00000␣␣␣␣␣␣␣␣␣␣␣␣␣--␣atom-label,␣Cartesian␣coordinates
End␣Of␣Basis␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣--␣end␣of␣basis␣set␣definition
```

The basis set label and the ECP libraries The label within the *ECP* library is given as input in the line following the keyword BASIS SET. The label defines either the valence basis set and core potential which is assigned to a frozen-core atom or the embedding potential which is assigned to an environmental froze-ion. Here, all the comments made about this label in the section **The basis set label and the basis set library** for all-electron basis sets stand, except for the following changes:

1. The identifier `type` must be `ECP` or `PP`.
2. The identifier `aux` specifies the kind of the potential. It is used, for instance, to choose between non-relativistic, Cowan-Griffin, or no-pair Douglas-Kroll relativistic core potentials
   (i.e. `Pt.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.16e-NR-AIMP.`
   or `Pt.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.16e-CG-AIMP.`
   or `Pt.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.18e-NP-AIMP.`)
   and to pick up one among all the embedding potentials available for a given ion
   (i.e. `F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.`
   or `F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-CsCaF3.`).
3. The identifier `contracted` is used here in order to produce the actual basis set out of the basis set included in the *ECP* library, which is a minimal basis set (in general contraction form) augmented with some polarization, diffuse, ... function. It indicates the number of `s, p, ...,` contracted functions in the actual basis set, the result being always a many-primitive contracted function followed by a number of primitives. As an example, `At.ECP.Barandiaran.13s12p8d5f.`**`3s4p3d2f`**`.17e-CG-AIMP.` will generate a (13,1,1/12,1,1,1/8,1,1/5,1) formal contraction pattern which is in this case a (13,1,1/12,1,1,1/7,1,1/5,1) real pattern. Other contraction patters should be input “Inline”.
4. The user is suggested to read carefully section 4.2.22 of the tutorials and examples manual before using the ECP utilities.

8.33.2 Numerical integration

Various Density Functional Theory (DFT) models can be used in *MOLCAS* . Energies and analytical gradients are available for all DFT models. In DFT the exact exchange present in HF theory is replaced by a more general expression, the exchange-correlation functional, which accounts for both the exchange energy, $\mathrm{E}_X[P]$ and the electron correlation energy ,$\mathrm{E}_C[P]$.

Description

We shall now describe briefly how the exchange and correlation energy terms look like. The functionals used in DFT are integrals of some function of the electron density and optionally the gradient of the electron density

$$E_X[P] = \int f(\rho_\alpha(r), \rho_\beta(r), \nabla\rho_\alpha(r), \nabla\rho_\beta(r))dr \tag{8.12}$$

The various DFT methods differ in which function, f, is used for $\mathrm{E}_X[P]$ and for $\mathrm{E}_C[P]$. In MOLCAS pure DFT methods are supported, together with hybrid methods, in which the exchange functional is a linear combination of the HF exchange and a functional integral of the above form. The latter are evaluated by numerical quadrature. In the `SEWARD` input the parameters for the numerical integration can be set up. In the `SCF` and `RASSCF` inputs the keywords for using different functionals can be specified. Names for the various pure DFT models are given by combining the names for the exchange and correlation functionals.

The DFT gradients has been implemented for both the fixed and the moving grid approach [146, 147, 148]. The latter is known to be translationally invariant by definition and is recommended in geometry optimizations.

Files

File	*Contents*
`RUNFILE`	The run file will contain the parameters defining and controlling the numerical integration.

Input

Below follows a description of the input to the numerical integration utility in the `SEWARD` input.

In addition to the keywords and the comment lines the input may contain blank lines. The input is always preceded by the following keyword in the `SEWARD` input

Grid Input

The first four character of the keywords are decoded while the rest are ignored. However, for a more transparent input we recommend the user to use the full keywords.

Compulsory keywords

Keyword	*Meaning*
END OF GRID-INPUT	This marks the end of the input to the numerical integration utility.

Optional keywords

Keyword	*Meaning*
GRID	It specifies the quadrature quality. The possible indexes that can follow are COARSE, SG1GRID, FINE, ULTRAFINE following the Gaussian98 convention. Default is FINE.
RQUAD	It specifies the radial quadrature scheme. Options are LOG3 (Mura and Knowles), BECKE (Becke) , MHL (Murray et a.), TA (Treutler and Ahlrichs, defined for H-Kr), and LMG (Lindh et al.), respectively. The default is MHL.
GGL	It activates the use of Gauss and Gauss-Legendre angular quadrature. Default is to use the Lebedev angular grid.
LEBEDEV	It turns on the Lebedev angular grid.
LOBATTO	It activates the use of Lobatto angular quadrature. Default is to use the Lebedev angular grid.
LMAX	It specifies the angular grid size. Default is 29.
NGRID	It specifies the maximum number of grid points to process at one instance. Default is 5500 grid points.
NOPRUNNING	It turns off the the angular prunning. Default is to prune.
NR	It is followed by the number of radial grid points. Default is 75 radial grid points.
FIXED GRID	Use a fixed grid in the evaluation of the gradient. This corresponds to using the grid to numerically evaluate the analytic gradient expression. Default is to use a moving grid.
MOVING GRID	Use a moving grid in the evaluation of the gradient. This correspond to evaluating the gradient of the numerical expression of the DFT energy. This is the default.
THRESHOLD	It is followed by a line containing the value for the the radial threshold. Default value is 1.0D-13.
T_X	Threshold for screening in the assembling of the density on the grid. Default value is 1.0D-18.
T_Y	Threshold for screening in the assembling of the integrals. Default value is 1.0D-11.
NOSCREENING	Turn of any screening in the numerical integration.

CROWDING	The crowding factor, according to MHL, used in the pruning of the angular grid close to the nuclei. Default value 3.0.

The `SCF` and `RASSCF` programs have their own keywords to decide which functionals to use in a DFT calculation.

Below follows an example of a DFT calculation with two different functionals.

```
␣&SEWARD
Basis␣set
H.3-21G.....
H1␣0.0␣␣0.0␣0.0
End␣of␣basis
Grid␣input
RQuad
Log3
nGrid
50000
GGL
lMax
26
Global
End␣of␣Input
␣&SCF
Occupations
1
KSDFT
LDA5
Iterations
1,␣1
End␣of␣input
␣&SCF
Occupations
1
KSDFT
B3LYP
Iterations
1,␣1
End␣of␣input
```

Using the Douglas–Kroll–Hess Hamiltonian

For all-electron calculations, the prefered way is to use the scalar-relativistic Douglas–Kroll–Hess (DKH) Hamiltonian, which, in principle, is available up to arbitrary order in MOLCAS; for actual calculations, however, it is advisable not to go beyond 4th order (the parameter settings of the implementation allow to run calculations up to 12th order in the external potential).

The arbitrary-order Hamiltonian is activated by setting

```
RXXPyy
```

somewhere in the `SEWARD` input, where the `XX` denotes the order of the DKH Hamiltonian in the external potential. I.e., for the standard 2nd-order Hamiltonian you may use `R02O`. Note in particular that the parametrization `P` does not affect the Hamiltonian up to fourth order. Therefore, as long as you run calculations with DKH Hamiltonians below 5th order you may use any symbol for the parametrization as they would all yield the same results.

The possible parametrizations `P` of the unitary transformation used in the DKH transformation supported by MOLCAS are:

`P=O`: Optimum parametrization (OPT)

`P=E`: Exponential parametrization (EXP)

`P=S`: Square-root parametrization (SQR)

`P=M`: McWeeny parametrization (MCW)

`P=C`: Cayley parametrization (CAY)

Note that the default correction for picture change artefacts on electric-field-like molecular properties like dipole moments or electric field gradients is a transformation of the four-component property operator up to 2nd order (with unitary transformations independent of the perturbation).

(Note: For `XX` ≥ 11 the values of some parameters in the file src/common/parameters.h have to be suitably increased. Only recommended for experts who do exactly know what they are doing!! For most cases `XX=10` is sufficient.)

Up to fourth order (`XX=04`) the DKH Hamiltonian is independent of the chosen paramterization. Higher-order DKH Hamiltonians depend slightly on the chosen paramterization of the unitary transformations applied in order to decouple the Dirac Hamiltonian.

For details on the arbitrary-order DKH Hamiltonians see [149] with respect to theory, [150] with respect to aspects of implementation, and [151] with respect to general principles of DKH.

For details on the different parametrizations of the unitary transformations see [152].

Douglas–Kroll–Hess transformed properties

As mentioned above, four-component molecular property operators need to be DKH transformed as well when going from a four-component to a two- or one-component description; the results do not coincide with the well-known correpsonding nonrelativistic expressions for a given property but are properly picture change corrected.

The transformation of electric-field-like molecular property operators can be carried out for any order smaller or equal to the order chosen for the scalar-relativistic DKH Hamiltonian. Usually, it is sufficient to carry out the property operator transformation up to 2nd order in the external electron–nucleus potential. Higher order transformations are in principle available to any order, but memory limitations set the limit of the current implementation to 9th or 10th order. However, property transformations of higher than 2nd order do require more unitary transformations than the DKH Hamiltonian of the same order does. Therefore, the DKH transformation of 3rd and higher order one-electron property matrices takes considerably more time.

In order to change the default transformation of order 2, you may concatenate the input for the DKH Hamiltonian by 2 more numbers specifying the order in the property,

```
RxxPyy
```

where `yy` denotes the order of the Hamiltonian starting with first order 01. The DKH transformation is then done automatically for all one-electron electric-field-like one-electron property matrices.

Also note that the current implementation of both the Hamiltonian and the property operators is carried out in the full, completely decontracted basis set of the molecule under consideration. The local nature of the relativistic contributions is not yet exploited and hence large molecules may require considerable computing time for all higher-order DKH transformations.

For details on the arbitrary-order DKH properties see [153] with respect to theory and [154] with respect to implementation aspects.

8.34 `SLAPAF`

Provided with the first order derivative with respect to nuclear displacements the program is capable to optimize molecular structures with or without constraints for minima or transition states. This will be achieved with a quasi-Newton approach in combination with 2nd ranks updates of the approximate Hessian or with the use of an analytic Hessian. Note that *if* an analytic Hessian is available on the ***`RUNFILE`*** then it will be used rather than the approximate Hessian generated by `SLAPAF`. **On completion of an optimization `SLAPAF` will automatically execute a single energy evaluation**.

8.34.1 Description

`SLAPAF` has three different ways in selecting the basis for the displacements during the optimization. The first format require user input (not recommended), whereas the two other options are totally black-boxed. The formats are:

1. the old format as in *MOLCAS*-3, which is user specified. The internal coordinates are here represented as linear combination of internal coordinates (such as bonds, angles, torsions, out of plane angles, Cartesian coordinates and separation of centers of mass) and the linear combinations are totally defined by user input. This format does also require the user to specify the Hessian (default a diagonal matrix). This option *allows* for frozen internal coordinates.
2. the second format is an automatic option which employs the Cartesian eigenvectors of the approximative Hessian (generated by the Hessian model functional [155]).
3. the third format (this is the recommend and default) is an automatic option which utilizes linear combinations of some curvilinear coordinates (stretches, bends, and torsions). This implementation has two variations. The first can be viewed as the conventional use of non-redundant internal coordinates [156, 157, 158]. The second variation is a force constant weighted (FCW) redundant space (the HWRS option) version of the former implementation [159].

All three formats of internal coordinates can be used in combinations with constraints on the molecular parameters or other type of constraints as for example energy differences.

The displacements are symmetry adapted and any rotation and translation if present is deleted. The relaxation is symmetry preserving.

8.34.2 Dependencies

`SLAPAF` depends on the results of `ALASKA` and also possibly on `MCKINELY` and `MCLR`.

8.34.3 Files

Input files

Apart from the standard input file `SLAPAF` will use the following input files.

File	*Contents*
RUNFILE	File for communication of auxiliary information. If an analytic Hessian is available on this file it will be used rather than the approximate Hessian generated by `SLAPAF`.
RUNFILE2	File for communication of auxiliary information of the ”ground state” in case of minimum energy cross point optimizations.
RUNOLD	File for communication of auxiliary information for reading an old Hessian matrix from a previous geometry optimization.

Output files

In addition to the standard output file `SLAPAF` will use the following output files.

File	*Contents*
RUNFILE	File for communication of auxiliary information.
RUNFILE2	File for communication of auxiliary information of the ”ground state” in case of minimum energy cross point optimizations.
MOLDEN	Molden input file for geometry optimization analysis.
MOLDEN2	Molden input file for minimum energy path (MEP).
MOLDEN3	Molden input file for energy path (MEP) of a Saddle TS optimization.
MOLDEN4	Molden input file for intrinsic reaction coordinate analysis of a TS.
MLDNFQ	Molden input file for harmonic frequency analysis.
UNSYM	ASCII file where all essential information, like geometry, Hessian normal modes and dipole derivatives are stored.
STRUCTURE	Output file with a statistics of geometry optimization convergence.

8.34.4 Input

`SLAPAF` will as standard provided with an energy and a corresponding gradient update the geometry (optimize). Possible update methods include different quasi-Newton methods. The program will also provide for updates of the Hessian. The program has a number of different variable metric methods available for the Hessian update. This section describes the input to the `SLAPAF` program. The input for each module is preceded by its name like:

```
␣&SLAPAF
```

Compulsory keywords

Keyword	*Meaning*
END OF INPUT	This marks the end of the input to the program.

Optional convergence control keywords

Keyword	*Meaning*
ITERATIONS	On the next lines follows the max number of iterations which will be allowed in the relaxation procedure. Default is 500 iterations, however, if MAXITER has been exported by the user this is the assumed default value.
THRSHLD	This keyword is followed by two real numbers on the next line which specifies the convergence criterion with respect to the energy change and the norm of the gradient. The defaults are 1.0D-6 and 3.0D-4.
BAKER	Activate convergence criterions according to Baker [160]. Default is to use the convergence criterions as in the Gaussian program [161].
MAXSTEP	This keyword is followed by the value which defines the seed of largest change of the internal coordinates which will be accepted. A change which is larger is reduced to the max value. The value is dynamically modified each iterations. The default value is 0.3 au or rad.
NOMAXSTEP	Activate that there should be made no modifications to the value of large changes of the internal coordinates. The default is to reduce large changes.

Optional coordinate selection keywords

Keyword	*Meaning*
CARTESIAN	Activate `SLAPAF` to use the eigenvectors of the approximative Hessian expressed in Cartesian as the definition of the internal coordinates. The default is to use the FCW non-redundant internal coordinates. The Hessian will be modeled by the Hessian Model Functional.

CONStraints	This marks the start of the definition of the constraints which the optimization is subject to. This section is always ended by the keyword End of Constraints. For a complete description of this keyword see the section 8.34.4. This option can be used in conjunction with any definition of the internal coordinates. This option will automatically turn off the line search. The defaults is to apply no constraints to the optimization.
INTErnal	This marks the start of the definition of the internal coordinates. This section is always ended by the keyword End of Internal. For a complete description of this keyword see the section 8.34.4. This option will also use a diagonal matrix as default for the Hessian matrix. The default is to use the FCW non-redundant internal coordinates.
HWRS	Use the force constant weighted (FCW) redundant space version of the nonredundant internal coordinates. This is the default. The Hessian will be modeled by the Hessian Model Functional.
NOHWrs	Disable the use of the force constant weighted redundant space version of the nonredundant internal coordinates. The default is to use the HWRS option. The Hessian will be modeled by the Hessian Model Functional.

Optional Hessian update keywords

Keyword	*Meaning*
NOUPdate	No update is applied to the Hessian matrix. Default is that the Broyden-Fletcher-Goldfarb-Shanno update is applied.
MEYEr	Activate update of the Hessian matrix according to Meyer's method [162, 163]. This method does not allow for any modifications of the proposed change of the geometry as suggested by the Hessian and the gradient. Default is that the Broyden-Fletcher-Goldfarb-Shanno update [164, 165, 166, 167] is applied.
BPUPdate	Activate update according to Broyden-Powell [168]. Default is that the Broyden-Fletcher-Goldfarb-Shanno update is applied.
BFGS	Activate update according to Broyden-Fletcher-Goldfarb-Shanno. This is the default.
MSP-update	Activate the Murtagh-Sargent-Powell update according to Bofill [169]. This update is preferred for the localization of transition states.
UORDer	Order the gradients and displacements vectors according to Schlegel prior to the update of the Hessian. Default is no reorder.

Optional optimization procedure keywords

Keyword	*Meaning*
NOLINE	Disable line search. Default is to use line search for minima.
RATIONAL	Activate geometry optimization using the restricted step Rational Functional optimization [170, 171], this is the default.
C1-DIIS	Activate geometry optimization using the C1-GDIIS method [172, 173, 174]. The default is to use the Rational Functional approach.
C2-DIIS	Activate geometry optimization using the C2-GDIIS method [175]. The default is to use the Rational Functional approach.
DXDX	This option is associated to the use of the C1- and C2-GDIIS procedures. This option will activate the computation of the so-called error matrix elements as $e = \delta x^\dagger \delta x$, where δx is the displacement vector.
DXG	This option is associated to the use of the C1- and C2-GDIIS procedures. This option will activate the computation of the so-called error matrix elements as $e = \delta x^\dagger g$, where δx is the displacement vector and g is the gradient vector.
GDX	See above.
GG	This option is associated to the use of the C1- and C2-GDIIS procedures. This option will activate the computation of the so-called error matrix elements as $e = g^\dagger g$, where g is the gradient vector. This is the default.
NEWTON	Activate geometry optimization using the standard quasi-Newton approach. The default is to use the Rational Functional approach.
RS-P-RFO	Activate RS-P-RFO [171] as default for TS-search. Default is RS-I-RFO.
TS	Keyword for optimization of transition states. This flag will activate the use of the mode following rational functional approach [176]. The mode to follow can either be the one with the lowest eigenvalue (if positive it will be changed to a negative value) or by the eigenvector which index is specified by the MODE keyword (see below). The keyword will also active the Murtagh-Sargent-Powell update of the Hessian and inactivate line search. This keyword will also enforce that the Hessian has the right index (i.e. one negative eigenvalue).
MODE	Specification of the Hessian eigenvector index, this mode will be followed by the mode following RF method for optimization of transition states. The keyword card is followed by a single card specifying the eigenvector index.
FINDTS	Enable a constrained optimization to release the constraints and locate a transition state if negative curvature is encountered and the gradient norm is below a specific threshold (see the GNRM option).

GNRM	Modified the gradient norm threshold associated with the FINDTS optiom. The actual threshold is specified on the subsequent line. The default value is 0.2.
MEP-search	Enable a minimum energy patch (MEP) search.
NMEP	Maximum number of points to find in a minimum energy patch search.
REFErence	The keyword is followed by a list of the symmetry unique coordinates (in a.u.) of the origin of the hyper sphere. The default origin is the structure of the first iteration.
GRADient of reference	The keyword is followed by a list of the gradient vector components. This keyword is compulsory when using the Transverse kind of constraint. The optimization is performed in a space orthogonal to the given vector.
CIOPtimization	The keyword is used to perform a conical intersection search. It has to be followed by a line containing the two roots of a previous `RASSCF` or `MS-CASPT2` run, between wich look for the intersection. Optionally, on the same line the keyword NUMERICAL can be specified to request a numerical procedure, default in case of `MS-CASPT2`. It will call the module `CIISCMNG`, which will produce the required gradients before returning to `SLAPAF`. See section 8.7 for help on how to write the input.
ISCOptimization	The keyword is used to perform an intersystem crossing search. It has to be followed by a line containing the two roots of two previous `RASSCF` or `CASPT2` runs, between wich look for the crossing. Optionally, on the same line the keyword NUMERICAL can be specified to request a numerical procedure, default in case of `CASPT2`. It will call the module `CIISCMNG`, which will produce the required gradients before returning to `SLAPAF`. See section 8.7 for help on how to write the input.
IRC	The keyword is used to perform an intrinsic reaction coordinate (IRC) analysis of a transition state structure. The analysis will follow the reaction patch forward and backward until the energy increase. The keyword require that the starting structure is that of a transition state and that the reaction vector is specified explicitly (check the keyword "REACtion vector") or implicitly can be found on *`RUNOLD`*.
REACtion vector	The keyword is followed by the reaction vector specified as the Cartesian vector components on each of the symmetry unique atoms.

Optional force constant keywords

Keyword	*Meaning*

SCHLEGEL	The approximate Hessian is computed according to Schlegel [177]. The default is to compute the approximate Hessian with the Hessian model functional [155].
OLDForce	The Hessian matrix is read from the file ***RUNOLD***. This Hessian is either an analytic or approximative Hessian updated by Slapaf. Note that for this option to work properly the type of internal coordinates must be the same!
FCONstant	Input of Hessian in iternal coordinates. There are two different syntaxes. 1. The keyword is followed by a line with the number of elements which will be set (observe that the update will preserve that the elements H_{ij} and H_{ji} are equal). The next lines will contain the value and the indices of the elements to be replaced. 2. The keyword if followed by the label Square or Triangular. The subsequent line specifies the rank of the Hessian. This is then followed by lines specifying the Hessian in square or lower triangular order.
XFCOnstant	Input of an external Hessian matrix in cartesian coordinates. The syntax is the same as for the FCONSTANT keyword.
NUMErical	This invokes as calculation of the force constant matrix by a two-point finite difference formula. The resulting force constant matrix is used for an analysis of the harmonic frequencies. **Observe** that in case of the use of internal coordinates defined as Cartesian coordinates that these has to be linear combinations which are free from translational and rotational components for the harmonic frequency analysis to be valid. **Alternative:** see keyword RowH in the section about Internal coordinates.
CUBIc	This invokes a calculation of the 2nd and the 3rd order force constant matrix by finite difference formula.
DELTa	This keyword is followed by a real number which defines the step length used in the finite differentiation. Default: 1.0D-2.
PRFC	The eigenvalues and eigenvectors of the Hessian matrix are printed. The internal coordinates definitions are also printed.
RHIDden	Define the hidden atoms selection radius in order to improve a QM/MM Hessian. It can be followed by Angstrom.

Optional miscellaneous keywords

Keyword	*Meaning*

CTOF	Coordinates TO Follow defines an internal coordinate whose values will be printed in the output during the optimization. Both the original and the new values will be printed. The keyword must be followed by the definition on the primitive coordinate.
RTRN	Max number of atoms for which bond lengths, angles and dihedral angles are listed, and the radius defining the maximum length of a bond follows on the next line. The latter is used as a threshold when printing out angles and dihedral angles. The length can be followed by BOHR or ANGSTROM which indicates the unit in which the length was specified, the default is BOHR. The default values are 15 and 3.0 au.
THERMOCHEMISTRY	Request frequencies to be computed followed by an user specified thermochemical analysis. The keyword must be followed by a line containing the Rotational Symmetry Number, a line containing the Pressure (in atm), and lines containing the Temperatures (in K) for which the thermochemistry will be calculated. The section is ended by the keyword "End of PT".

Example: A complete set of input decks for a CASSCF geometry optimization. These are the input decks for the optimization of the enediyne molecule.

```
␣&SEWARD
Title
Enediyne␣MCSCF␣structure
Symmetry
x␣z
Basis␣set
C.ANO...5s4p2d.
C1␣␣␣␣␣␣␣␣␣␣␣␣␣1.2869761127␣␣␣␣␣␣␣␣2.0799281025␣␣␣␣␣␣␣␣␣.0000000000
C2␣␣␣␣␣␣␣␣␣␣␣␣␣2.8355091288␣␣␣␣␣␣␣␣-.1380881195␣␣␣␣␣␣␣␣␣.0000000000
C3␣␣␣␣␣␣␣␣␣␣␣␣␣4.1954709187␣␣␣␣␣␣␣-1.9656839604␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
Basis␣set
H.ANO...3s2p.
H1␣␣␣␣␣␣␣␣␣␣␣␣␣2.2478721352␣␣␣␣␣␣␣␣3.8639049616␣␣␣␣␣␣␣␣␣.0000000000
H2␣␣␣␣␣␣␣␣␣␣␣␣␣5.3554366293␣␣␣␣␣␣␣-3.5799988030␣␣␣␣␣␣␣␣␣.0000000000
End␣of␣basis
End␣of␣input
␣&SCF
Title
Enediyne
ITERATIONS
␣30
Occupied
9␣8␣2␣1
Thresholds
␣1.d-8␣␣.5d-8
IVO
End␣of␣input
␣&RASSCF
Lumorb
NactEl
12␣0␣0
Spin
1
Inactive
7␣7␣0␣0
Ras2
```

```
3␣3␣3␣3
Iterations
50␣50
CiRoot
1␣1
1
Thrs
␣1.0e-08␣1.0e-05␣1.0e-05
Symmetry
␣1
End␣of␣input
␣&ALASKA
End␣of␣input
␣&SLAPAF
Iterations
20
End␣of␣input
```

Example: Thermochemistry for an asymmetric top (Rotational Symmetry Number = 1), at 1.0 atm and 273.15, 298.15, 398.15 and 498.15 K.

```
␣&SLAPAF
THERmochemistry
␣1
␣1.0
␣273.15
␣298.15
␣398.15
␣498.15
End␣of␣PT
End␣of␣input
```

Definition of internal coordinates or constraints

The input section defining the internal coordinates always start with the keyword INTERNAL COORDINATES and the definition of the constraints starts with the keyword CONSTRAINTS.

The input is always sectioned into two parts where the first section defines a set of primitive internal coordinates and the second part defines the actual internal coordinates as any arbitrary linear combination of the primitive internal coordinates that was defined in the first section. In case of constraints the second part does also assign values to the constraints.

In the first section we will refer to the atoms by their atom label (`SEWARD` will make sure that there is no redundancy). In case of symmetry one will have to augment the atom label with a symmetry operation in parenthesis in order to specify a symmetry related center. Note that the user only have to specify distinct internal coordinates (`ALASKA` will make the symmetry adaptation).

In the specification below *rLabel* is a user defined label with no more than 8 (eight) characters. The specifications atom1, atom2, atom3, and atom4 are the unique atom labels as specified in the input to `SEWARD`.

The primitive internal coordinates are defined as

- ***rLabel* = Bond atom1 atom2** — a primitive internal coordinate *rLabel* is defined as the bond between center atom1 and atom2.

- ***rLabel* = Angle atom1 atom2 atom3** — a primitive internal coordinate *rLabel* is defined as the angle between the bonds formed from connecting atom1 to atom2 and connecting atom2 to atom3.

- ***rLabel* = LAngle(1) atom1 atom2 atom3** — a primitive internal coordinate *rLabel* is defined as the linear angle between the bonds formed from connecting atom1 to atom2 and connecting atom2 to atom3. To define the direction of the angle the following procedure is followed.

  1. – *the three centers are linear*,
     (a) form a reference axis, **R1**, connecting atom1 and atom3,
     (b) compute the number of zero elements, *nR*, in the reference vector,
        i. – *nR=0*, a first perpendicular direction to the reference axis is formed by

        $$\mathbf{R2} = (R1x, R1y, -R1z)$$

        followed by the projection

        $$\mathbf{R2} = \mathbf{R2} - \frac{\mathbf{R2}\cdot\mathbf{R1}}{\mathbf{R1}\cdot\mathbf{R1}}\mathbf{R1}.$$

        The second perpendicular direction completes the right-handed system.
        ii. – *nR=1*, a first perpendicular direction to the reference axis is defined by setting the element in **R2** corresponding to the zero entry in **R1** to unity. The second perpendicular direction completes the right-handed system.
        iii. – *nR=2*, a first perpendicular direction to the reference axis is defined by setting the element corresponding to the first zero entry in **R1** to unity. The second perpendicular direction completes the right-handed system.
  2. – *the three centers are nonlinear*, the first perpendicular direction is the one which is in the plane formed by atoms atom1, atom2, and atom3. The second perpendicular direction is taken as the direction perpendicular to the same plane.

  The direction of the bend for **LAngle(1)** is taken in the direction of the first perpendicular direction, etc.

- ***rLabel* = LAngle(2) atom1 atom2 atom3** — a primitive internal coordinate *rLabel* is defined as the linear angle between the bonds formed from connecting atom1 to atom2 and connecting atom2 to atom3. The definition of the perpendicular directions is as described above. The direction of the bend for **LAngle(2)** is taken in the direction of the second perpendicular direction.

- ***rLabel* = Dihedral atom1 atom2 atom3 atom4** — a primitive internal coordinate *rLabel* is defined as the angle between the planes formed of atom1, atom2 and atom3, and atom2, atom3 and atom4, respectively.

- ***rLabel* = OutOfP atom1 atom2 atom3 atom4** — a primitive internal coordinate *rLabel* is defined as the angle between the plane formed by atom2, atom3, and atom4 and the bond formed by connecting atom1 and atom4.

- ***rLabel* = Dissoc (n1+n2) atom1 atom2 atom3 ... atomN** — a primitive internal coordinate *rLabel* is defined as the distance between the center of masses of two sets of centers. The first center has n1 members and the second has n2. The input contains the labels of the atoms of the first group followed immediately by the labels of the second group. This option is not available for constraints.

- ***rLabel* = Cartesian i atom1** — a primitive internal coordinate *rLabel* is defined as the pure Cartesian displacement of the center labeled atom1. The label i is selected to x, y, or z to give the appropriate component.
- ***rLabel* = Ediff** — an energy difference. The information of the second state is provided on RUNFILE2. This is only used in constrained optimization in which intersections or conical intersections are located.
- ***rLabel* = Sphere** — the radius of the hypersphere defined by two different molecular structures (the origin is the first structure) in relative mass-weighted coordinates. This is only used in constrained optimization in which minimum reaction paths (MEP) or intrinsic reaction coordinate (IRC) paths are followed. The units of the radius is in mass-weighted coordinates divided with the square root of the total mass of the molecule.
- ***rLabel* = Transverse** — a level of “orthogonality”. This is used to perform an optimization in a space orthogonal to a given vector. Recommended value 0.0. Requires usage of GRAD keyword.

The second section starts with the label VARY or in the case of constraints with the label VALUES.

In case of a definition of **internal coordinates** in this section the user specifies all symmetric internal coordinates excluding translation and rotation using a list of expressions like

label = f1 *rLabel1* + f2 *rLabel2* + ...

which defines an internal coordinate *label* as the linear combination of the primitive internal coordinates *rLabel1*, *rLabel2*, ... with the coefficients f1, f2, ..., respectively. If the internal coordinate just corresponds to the primitive internal coordinate, the same label can be used

label

If some internal coordinates are chosen to be fixed they should be defined after the label FIX. The fixed internal coordinate are defined with expressions as in the section VARY. Observe: using expression can introduce linear dependence and/or undefined nuclear coordinates, so use with care.
For the internal coordinates defined after VARY (and FIX, if present) a numerical estimation of rows and columnns of the hessian matrix can be performed. The *label* of internal coordinates (max 10) must be specified after keyword ROWH. Keywords NUMERICAL and ROWH are mutually exclusive.

In case of a definition of **constraints** the sections contains either a direct reference to a *rLabel* as in

rLabel = *rValue* [Angstrom,Degrees]

or one can also use expressions like

f1 *rLabel1* + f2 *rLabel2* + ... = *Value* [Angstrom,Degrees]

where *rValue* is the desired value of the constraint in au or rad.

Example: A definition of user specified internal coordinates of benzene. The molecule is in D_{6h} and since *MOLCAS* only uses up to D_{2h} the FIX option is used to constrain the relaxation to the higher point group. **Observe** that this will only restrict the nuclear coordinates to D_{6h}. The electronic wavefunction, however, can have lower symmetry.

```
Internal␣coordinates
r1␣=␣Bond␣C1␣C2
r2␣=␣Bond␣C1␣H1
r3␣=␣Bond␣C2␣H2
r4␣=␣Bond␣C2␣C2(x)
f1␣=␣Angle␣H1␣C1␣C2
f2␣=␣Angle␣H2␣C2␣C1
Vary
a␣=␣1.0␣r1␣+␣␣1.0␣r4
b␣=␣1.0␣r2␣+␣␣1.0␣r3
c␣=␣1.0␣f1␣+␣␣1.0␣f2
Fix
a␣=␣1.0␣r1␣+␣-1.0␣r4
b␣=␣1.0␣r2␣+␣-1.0␣r3
c␣=␣1.0␣f1␣+␣-1.0␣f2
End␣of␣Internal
```

Example: A input for the optimization of water constraining the structure to be linear at convergence.

```
>>>␣DO␣WHILE␣<<<
␣&SEWARD
Title
␣H2O␣geom␣optim,␣using␣the␣ANO-S␣basis␣set.
Pkthre
1.0D-11
Basis␣set
H.ANO-S...1s.
H1␣␣␣␣␣␣1.43354233␣␣␣␣.00000000␣␣␣␣.95295406
H2␣␣␣␣␣-1.43354233␣␣␣␣.00000000␣␣␣␣.95295406
End␣of␣basis
Basis␣set
O.ANO-S...2s1p.
O␣␣␣␣␣␣␣␣.00000000␣␣␣␣.00000000␣␣␣␣.00000000
End␣of␣basis
End␣of␣input
␣&SCF
ITERATIONS
␣40
Occupied
␣5
End␣of␣input
␣&SLAPAF
Iterations
␣15
Constraints
a1␣=␣langle(1)␣H1␣O␣H2
Values
a1␣=␣180.00␣degrees
End␣of␣Constraints
End␣of␣input
>>>␣END␣DO␣<<<
```

Example: A complete set of input decks for a UHF transition structure geometry optimization of an identity hydrogen transfer reaction (HO + H_2O -> H_2O + OH).

```
>>>␣DO␣WHILE␣<<<
␣&SEWARD
ZMAT
O.STO-3G....
```

```
H.STO-3G....
H1
X2␣␣␣␣1␣␣1.0
O3␣␣␣␣1␣␣1.15␣␣␣␣2␣␣␣92.
O4␣␣␣␣1␣␣1.15␣␣␣␣2␣␣␣92.␣␣␣␣␣3␣␣180.
H5␣␣␣␣3␣␣0.98␣␣␣␣4␣␣105.4␣␣␣␣2␣␣120.
H6␣␣␣␣4␣␣0.98␣␣␣␣3␣␣105.4␣␣␣␣2␣␣120.
End␣of␣input
␣&SCF
UHF
End␣of␣input
␣&SLAPAF
TS
PRFC
Internal
OO4␣␣=␣Bond␣O3␣O4
OH5␣␣=␣Bond␣H5␣O3
OH6␣␣=␣Bond␣H6␣O4
OOH5␣=␣Angle␣O4␣O3␣H5
OOH6␣=␣Angle␣O3␣O4␣H6
DH6␣␣=␣Dihedral␣H6␣O4␣O3␣H5
OH1␣␣=␣Bond␣O3␣H1
HOH1␣=␣Angle␣H5␣O3␣H1
DH1␣␣=␣Dihedral␣O4␣H5␣O3␣H1
Vary
rOH1
rOO4
rOH5
rOH6
aOOH5
aOOH6
tDH6
aHOH1
tDH1
RowH
rOH1
End␣of␣Internal
Iterations
␣20
End␣of␣input
>>>␣END␣DO␣<<<
```

8.35 VIBROT

The program `VIBROT` is used to compute a vibration-rotation spectrum for a diatomic molecule, using as input a potential computed over a grid. The grid should be dense around equilibrium (recommended spacing 0.05 au) and should extend to large distance (say 50 au) if dissociation energies are computed.

The potential is fitted to an analytical form using cubic splines. The ro-vibrational Schrödinger equation is then solved numerically (using Numerov's method) for one vibrational state at a time and for a number of rotational quantum numbers as specified by input. The corresponding wave functions are stored on file *`VIBWVS`* for later use. The ro-vibrational energies are analyzed in terms of spectroscopic constants. Weakly bound potentials can be scaled for better numerical precision.

The program can also be fed with property functions, such as a dipole moment curve. Matrix elements over the ro-vib wave functions for the property in question are then computed. These results can be used to compute IR intensities and vibrational averages of different properties.

`VIBROT` can also be used to compute transition properties between different electronic states. The program is then run twice to produce two files of wave functions. These files are used as input in a third run, which will then compute transition matrices for input properties. The main use is to compute transition moments, oscillator strengths, and lifetimes for ro-vib levels of electronically excited states. The asymptotic energy difference between the two electronic states must be provided using the ASYMptotic keyword.

8.35.1 Dependencies

The `VIBROT` is free-standing and does not depend on any other program.

8.35.2 Files

Input files

The calculation of vibrational wave functions and spectroscopic constants uses no input files (except for the standard input). The calculation of transition properties uses *`VIBWVS`* files from two preceding `VIBROT` runs, redefined as *`VIBWVS1`* and *`VIBWVS2`*.

Output files

`VIBROT` generates the file *`VIBWVS`* with vibrational wave functions for each v and J quantum number, when run in the wave function mode. If requested `VIBROT` can also produce a file *`VIBPLT`* with the fitted potential and property functions for later plotting.

8.35.3 Input

This section describes the input to the `VIBROT` program in the *MOLCAS* program system. The program name is

```
␣&VIBROT
```

Keywords

The first four characters are decoded, while the rest are ignored. Numerical input which follows the keyword is always in free format. The first keyword to `VIBROT` is an indicator for the type of calculation that is to be performed. Two possibilities exist:

Keyword	*Meaning*
ROVIBRATIONAL SPECTRUM	`VIBROT` will perform a vib-rot analysis and compute spectroscopic constants.
TRANSITION MOMENTS	`VIBROT` will compute transition moment integrals using results from two previous calculations of the vib-rot wave functions.

Note that only one of the above keywords can be used in a single calculation. If none is given the program will only process the input section.

After this first keyword follows a set of keywords, which are used to specify the run. Most of them are optional. **Note:** with keyword TRANSITION MOMENTS only OBSERVABLE and END OF INPUT are valid keywords.

The compulsory keywords are:

Keyword	*Meaning*
ATOMS	All isotope masses are stored in the program. You may introduce your own masses by giving a negative integer value to the isotope number (one of them or both). The masses (in ^{12}C units) are then read on the next (or next two) line(s). The isotopes of hydrogen can be given as H, D, or T.
POTENTIAL	Gives the potential as an arbitrary number of lines. Each line contains a bond distance (in au) and an energy value (in au). A plot file of the potential is generated if the keyword PLOT is added after the last energy input. One more line should then follow specifying the start and end value for the internuclear distance and the distance between adjacent plot points. This input must only be given together with the keyword ROVIBRATIONAL SPECTRUM.

In addition you may want to specify some of the following optional input:

Keyword	*Meaning*
TITLE	An arbitrary number (less than 10) title cards follows on the next lines.
GRID	The next lines give the number of grid points used in the numerical solution of the radial Schrödinger equation. The default value is 199. The maximum value that can be used is 499.
RANGE	The next line contains to distances Rmin and Rmax (in au) specifying the range in which the vibrational wave functions will be computed. The default values are 1.0 and 5.0 au. Note that these values most often have to be given as input since they vary considerably from one case to another. If the range specified is too small, the program will give a message informing the user that the vibrational wave function is large outside the integration range.
VIBRATIONAL	The next line specifies the number of vibrational quanta for which the wave functions and energies are computed. Default value is 3.
ROTATIONAL	The next line specifies the range of rotational quantum numbers. Default values are 0 to 5. If the orbital angular momentum quantum number (m_ℓ) is non zero, the lower value will be adjusted to m_ℓ if the start value given in input is smaller than m_ℓ.

ORBItal	The next line specifies the value of the orbital angular momentum (0,1,2, etc). Default value is zero.
SCALe	This keyword is used to scale the potential, such that the binding energy is 0.1 au. This leads to better precision in the numerical procedure and is strongly advised for weakly bound potentials. No additional input.
NOSPectroscopic	Only the wave function analysis will be carried out but not the calculation of spectroscopic constants.
OBSErvable	This keyword indicates the start of input for radial functions of observables other than the energy, for example the dipole moment function. The next line gives a title for this observable. An arbitrary number of input lines follows. Each line contains a distance and the corresponding value for the observable. As for the potential this input can also end with the keyword Plot, to indicate that a file of the function for later plotting is to be constructed. The next line then contains the minimum and maximum R-values and the distance between adjacent points. When this input is given with the top keyword Vibrational spectrum the program will compute matrix elements for vibrational wave functions of the current electronic state. Transition moment integrals are instead obtained when the top keyword is Transition moments. In the latter case the calculation becomes rather meaningless if this input is not provided. The program will then only compute the overlap integrals between the vibrational wave functions of the two states. The keyword Observable can be repeated up to ten times in a single run. All observables should be given in atomic units.
STEP	The next line gives the starting value for the energy step used in the bracketing of the eigenvalues. The default value is 0.004 au (88cm-1). This value must be smaller than the zero point vibrational energy of the molecule.
ASYMptotic	The next lines specifies the asymptotic energy difference between two potential curves in a calculation of transition matrix elements. The default value is zero atomic units.

Input example

```
␣&VIBROT
RoVibrational␣spectrum
Title
␣Vib-Rot␣spectrum␣for␣FeNi
Atoms
0␣Fe␣0␣Ni
Potential
1.0␣-0.516768
1.1␣-0.554562
 :
 :
```

```
Plot
1.0␣10.0␣0.1
Grid
150
Range
1.0␣10.0
Vibrations
10
Rotations
2␣10
Orbital
2
Observable
Dipole␣Moment
1.0␣0.102354
1.1␣0.112898
:
:
Plot
1.0␣10.0␣0.1
```

Comments: The vibrational-rotation spectrum for FeNi will be computed using the potential curve given in input. The 10 lowest vibrational levels will be obtained and for each level the rotational states in the range J=2 to 10. The vib-rot matrix elements of the dipole function will also be computed. A plot file of the potential and the dipole function will be generated. The masses for the most abundant isotopes of Fe and Ni will be selected.

8.36 The Basis Set Libraries

The basis sets library contains both all-electron and effective core potentials. They will be briefly described below and we refer to the publications for more details. The user can also add new basis sets to the basis directory and the structure of the file will therefore be described below.

Dummy atoms

Note that to use dummy atoms the user should employ the basis set label "X....". This will signify centers associated with no charge and no basis functions.

The All Electron Basis Set Library

The basis set library of MOLCAS contains an extensive set of basis sets both segmented and generally contracted. The files in the basis directory are named in upper case after the basis type label (see below). Three sets of generally contracted basis sets have been especially designed for MOLCAS. They are based on the Atomic Natural Orbital (ANO) concept and are labeled ANO-X (X=S, L, or RCC). They have been designed to give a balanced bescription of the atoms in ground, excited, and ionized states. A more detailed description of these basis sets is given below. A fourth basis set, which is especially designed for the calculation of electric properties of molecules (POL) will also be described.

In addition to this, an subset of segmented standard basis sets are included, for example, STO-3G, 3-21G 4-31G, 6-31G, 6-31G*, 6-31G**, cc-pVXZ (X=D,T,Q), and aug-cc-pVXZ

(X=D,T). In addition, the library also contains different variants of the Turbomole RI basis sets. For additional all electron basis set we recommend a visit to the EMSL Gaussian Basis Set Order Form
(http://www.emsl.pnl.gov/forms/basisform.html). All basis sets are stored in the directory basis_library. The different types of available basis sets can be found in the file basistype.tbl in this directory. Aliases for the names are listed in the file basis.tbl. However, the best way to find out which basis sets are available is to issue the command "molcas help basis X" where X is the atom. Note that a short hand notation can be used for most basis sets: for example ANO-L-VTZP will give a basis set of valence triple zeta accuracy with polarization functions.

Small ANO basis sets — ANO-S The smallest of the Atomic Natural Orbital (ANO) basis sets are available for the atoms H–Kr. They have been constructed as eigenfunctions of a density matrix averaged over several electronic configurations. The ground state of the atom was included for all atoms, and dependent on the particular atom one or more of the following states were included: valence excited states, ground state for the anion and ground state for the cation. The density matrices were obtained by the SCF, SDCI or MCPF methods for 1 electron, 2 electron and many electron cases respectively. The emphasis have been on obtaining good structural properties such as bond-lengths and -strengths with as small contracted sets as possible. The quality for electric properties such as polarizabilities have been sacrificed for the benefit of the properties mentioned above. See [106] for further discussions. These basis sets are recommended for large molecules where the more extended ANO-L basis sets require to much computational times. One should, however, remember that for a given contraction it is only the time needed to generate the integrals (or Cholesky vectors) that is affected and it is usually preferred to use the more accurate ANO-L (or ANO-RCC) basis sets.

For information about the primitive basis set we refer to the library. The maximum number of ANO's given in the library is:

- 4s3p for H–He.
- 6s4p3d for Li–Be.
- 7s6p3d for B–Ne.
- 7s5p3d for Na–Mg.
- 7s7p4d for Al–Ar.
- 7s7p4d for K–Ca.
- 8s7p7d4f for Sc–Zn.
- 9s9p5d for Ga–Kr.

However, such contractions are unnecessarily large. Almost converged results (compared to the primitive sets) are obtained with the basis sets:

- 3s2p for H–He.

- 4s3p2d for Li–Ne.
- 5s4p3d for Na–Ar.
- 6s5p4d for K–Ca.
- 7s5p4d3f for Sc–Zn.
- 6s5p4d for Ga–Kr.

The results become more approximate below the DZP size:

- 2s1p for H–He.
- 3s2p1d for Li–Ne.
- 4s3p2d for Na–Ar.
- 5s4p3d for K–Ca.
- 6s4p3d2f for Sc–Zn.
- 5s4p3d for Ga–Kr.

Large ANO basis sets — ANO-L The large ANO basis sets for atoms H–Zn, excluding K and Ca, have been constructed by averaging the corresponding density matrix over several atomic states, positive and negative ions and the atom in an external electric field [103, 104, 105]. The different density matrices have been obtained from correlated atomic wave functions. Usually the SDCI method has been used. The exponents of the primitive basis have in some cases been optimized. The contracted basis sets give virtually identical results as the corresponding uncontracted basis sets for the atomic properties, which they have been optimized to reproduce. The design objective has been to describe the ionization potential, the electron affinity, and the polarizability as accurately as possible. The result is a well balanced basis set for molecular calculations.

For information about the primitive basis set we refer to the library. The maximum number of ANO's given in the library is:

- 6s4p3d for Hydrogen.
- 7s4p3d for Helium.
- 7s6p4d3f for Li–Be.
- 7s7p4d3f for B–Ne.
- 7s7p5d4f for Na–Ar.
- 8s7p6d5f4g for Sc–Zn

However, such contractions are unnecessarily large. Almost converged results (compared to the primitive sets) are obtained with the VQZP basis sets:

- 3s2p1d for H–He.
- 5s4d3d2f for Li–Ne.
- 6s5p4d3f for Na–Ar.
- 7s6p5d4f3g for Sc–Zn

The results become more approximate below the size:

- 3s2p for H–He.
- 4s3p2d for Li–Ne
- 5s4p2d for Na–Ar.
- 6s5p4d3f for Sc–Zn

It is recommended to use at least two polarization (3d/4f) functions, since one of them is used for polarization and the second for correlation. If only one 3d/4f-type function is used one has to decide for which purpose and adjust the exponents and the contraction correspondingly. Here both effects are described jointly by the two first 3d/4f-type ANO's (The same is true for the hydrogen 2p-type ANO's). For further discussions regarding the use of these basis sets we refer to the literature [103, 104, 105].

Relativistic ANO basis sets — ANO-RCC Extended relativistic ANO-type basis sets are available for the atoms H–Cm. These basis sets have been generated using the same principles as described above for the ANO-L basis sets with the difference that the density matrices have been computed using the CASSCF/CASPT2 method. The basis have been contracted using the Douglas-Kroll Hamiltonian and should therefore only be used in calculations where scalar relativistic effects are included. Seward will automatically recognize this and turn on the DK option when these basis sets are used [2, 3, 4, 5]. The basis sets contain functions for correlation of the semi-core electrons. The new basis sets are called ANO-RCC. More details about the construction and performance is given in the header for each basis set in the ANO-RCC library. Basis sets are available for all atoms up to Cm.

Scalar relativistic effect become important already in the second row of the periodic systems. It is therefore recommended to use these basis sets instead of ANO-L in all calculations.

For information about the primitive basis set we refer to the library. The maximum number of ANOs given in the library is:

- 6s4p3d1f for Hydrogen.
- 7s4p3d2f for Helium.
- 8s7p4d2f1g for Li–Be.
- 8s7p4d3f2g for Be–Ne.
- 17s12p5d4f for Na.
- 9s8p5d4f for Mg–Al.

- 8s7p5d4f2g for Si–Ar
- 10s9p5d3f for K
- 10s9p6d2f for Ca
- 10s10p8d6f4g2h for Sc–Zn
- 9s8p6d4f2g for Ga–Kr
- 10s10p5d4f for Rb–Sr
- 10s9p8d5f3g for In–Xe
- 12s10p8d4f for Cs–Ba
- 11s10p8d5f3g for La
- 12s11p8d7f4g2h for Ce–Lu
- 11s10p9d8f4g2h for Hf–Hg
- 11s10p9d6f4g for Tl–Rn
- 12s11p8d5f for Fr-Ra
- 13s11p10d8f6g3h for Ac-Pa
- 12s10p9d7f5g3h for U-Cm

However, such contractions are unnecessarily large. Almost converged results (compared to the primitive sets) are usually obtained with basis sets of QZP quality. You can get a feeling for the convergence from the test results presented in the header of each basis set in the library. One should also remember that larger basis sets are needed for the correlation of semi-core electrons.

Below is a list of the core electrons correlated for each atom.

Li–B: 1s
C–Ne: No core correlation
Na: 2s,2p
Mg–Al: 2p
Si–Ar: No core correlation
K: 3s,3p
Ca–Zn: 3p
Ga–Ge: 3d
As–Kr: No core correlation
Rb–Sr: 4p
In–Xe: 4d
Cs–Ba: 5p
La–Lu: 5s,5p
Hf–Re: 4f,5s,5p
Os–Hg: 5s,5p
Tl–Rn: 5d
Fr–Ra: 6p
Ac-Cm: 6s,6p

Basis set label in input:
The general label is given as for the other ANO basis sets:
Atom.ano-rcc...contracted set. (Note the last dot!). A short hand notation is also possible: *Atom.ANO-RCC-label*, where *label* is one of MB,VDZ,VDZP,VTZP, or VQZP. A translation between the two possibilities can be found in file: `$MOLCAS/basis_library/basis.tbl`

Polarized basis sets The so-called polarized basis sets are purpose oriented, relatively small GTO/CGTO sets devised for the purpose of accurate calculations of dipole electric properties of polyatomic molecules [178, 179, 180, 181, 182]. For each row of the periodic table the performance of the basis sets has been carefully examined in calculations of dipole moments and dipole polarizabilities of simple hydrides at both the SCF and correlated levels of approximation [178, 179, 180, 181, 182]. The corresponding results match within a few percent the best available experimental data. Also the calculated molecular quadrupole moments turn out to be fairly close to those computed with much larger basis sets. According to the present documentation the polarized basis GTO/CGTO sets can be used for safe accurate predictions of molecular dipole moments, dipole polarizabilities, and also molecular quadrupole moments by using high-level correlated computational methods. The use of the polarized basis sets has also been investigated in calculations of weak intermolecular interactions. The interaction energies, corrected for the basis set superposition effect (BSSE), which is rather large for these basis sets, turn out to be close to the best available data. In calculations for molecules involving the 4th row atoms, the property data need to be corrected for the relativistic contribution. The corresponding finite perturbation facility is available [183, 184].

It is recommended to use these basis sets with the contraction given in the library. It is of course possible to truncate them further, for example by deleting some polarization functions, but this will lead to a deterioration of the computed properties.

Below is a list of the basis sets available in this class.

```
H.Pol.Sadlej.6s4p.3s2p.
Li.Pol.Sadlej.10s6p4d.5s3p2d.
Be.Pol.Sadlej.10s6p4d.5s3p2d.
C.Pol.Sadlej.10s6p4d.5s3p2d.
N.Pol.Sadlej.10s6p4d.5s3p2d.
O.Pol.Sadlej.10s6p4d.5s3p2d.
F.Pol.Sadlej.10s6p4d.5s3p2d.
Na.Pol.Sadlej.14s10p4d.7s5p2d.
Mg.Pol.Sadlej.14s10p4d.7s5p2d.
Si.Pol.Sadlej.14s10p4d.7s5p2d.
P.Pol.Sadlej.14s10p4d.7s5p2d.
S.Pol.Sadlej.14s10p4d.7s5p2d.
Cl.Pol.Sadlej.14s10p4d.7s5p2d.
K.Pol.Sadlej.15s13p4d.9s7p2d.
Ca.Pol.Sadlej.15s13p4d.9s7p2d.
Ge.Pol.Sadlej.15s12p9d.9s7p4d.
As.Pol.Sadlej.15s12p9d.9s7p4d.
Se.Pol.Sadlej.15s12p9d.9s7p4d.
Br.Pol.Sadlej.15s12p9d.9s7p4d.
```

```
Rb.Pol.Sadlej.18s15p10d.11s9p4d.
Sr.Pol.Sadlej.18s15p10d.11s9p4d.
Sn.Pol.Sadlej.19s15p12d4f.11s9p6d2f.
Sb.Pol.Sadlej.19s15p12d4f.11s9p6d2f.
Te.Pol.Sadlej.19s15p12d4f.11s9p6d2f.
I.Pol.Sadlej.19s15p12d4f.11s9p6d2f.
```

Structure of the all electron basis set library

The start of a given basis set in the library is given by the line

```
/label
```

where "label" is the basis set label, as defined below in the input description to `SEWARD`. Then follows two lines with the appropriate literature reference for that basis set. These cards are read by `SEWARD` and must thus be included in the library, and may not be blank. Next is a set of comment lines, which begin with an asterisk in column 1, giving some details of the basis sets. A number of lines follow, which specifies the basis set:

1. Charge of the atom and the highest angular momentum. For each angular momentum (l) then follows.
2. Number of primitives and contracted functions for angular momentum l (must be identical to those given in the basis set label) .
3. Exponents of the primitive functions .
4. The contraction matrix (with one CGTO per column). Note that all basis sets are given in the generally contracted format, even if they happen to be segmented. Note that the number of CGTOs must correspond to the data given in the label .

The following is an example of an entry in a basis set library.

```
* This is the Huzinaga 5s,2p set contracted to 3s,2p        -- Comment
* according to the Dunning paper.                           -- Comment
/H.TZ2P.Dunning.5s2p.3s2p.                                  -- Label
Exponents  : S. Huzinaga, J. Chem. Phys., 42, 1293(1965).   -- First ref line
Coefficients: T. H. Dunning, J. Chem. Phys., 55, 716(1971). -- Second ref line
 1.0 1                                                      -- Charge, sp
 5 3                                                        -- 5s->3s
 52.56 7.903 1.792 0.502 0.158                              -- s-exponents
 0.025374  0.0  0.0       -- contr. matrix
 0.189684  0.0  0.0       -- contr. matrix
 0.852933  0.0  0.0       -- contr. matrix
 0.0    1.0  0.0       -- contr. matrix
 0.0    0.0  1.0       -- contr. matrix
 2  2                                                       -- 2p->2p
 1.5  0.5                                                   -- p-exponents
 1.0 0.0             -- contr. matrix
 0.0 1.0             -- contr. matrix
```

The ECP Library

MOLCAS is able to perform *effective core potential* (ECP) calculations and *embedded cluster* calculations. In ECP calculations, only the *valence* electrons of a molecule are explicitly handled in a quantum mechanical calculation, at a time that the *core* electrons are kept frozen and are represented by ECP's. (An example of this is a calculation on HAt in which only the 5d, 6s and 6p electrons of Astatine and the one of Hydrogen are explicitly considered.) Similarly, in *embedded cluster* calculations, only the electrons assigned to a piece of the whole system (the *cluster*) are explicitly handled in the quantum mechanical calculation, under the assumption that they are the only ones relevant for some local properties under study; the rest of the whole system (the *environment*) is kept frozen and represented by embedding potentials which act onto the *cluster*. (As an example, calculations on a TlF_{12}^{11-} cluster embedded in a frozen lattice of $KMgF_3$ can be sufficient to calculate spectroscopical properties of Tl^+-doped $KMgF_3$ which are due to the Tl^+ impurity.)

In order to be able to perform ECP calculations in molecules, as well as *embedded cluster* calculations in ionic solids, with the Ab Initio Model Potential method (AIMP) [185, 186, 187, 188, 189, 190] MOLCAS is provided with the library ECP which includes nonrelativistic and relativistic *core* ab initio model potentials and *embedding* ab initio model potentials representing both complete-cations and complete-anions in ionic lattices [186, 191].

Before we continue we should comment a little bit on the terminology used here. Strictly speaking, ECP methods are all that use the frozen-core approximation. Among them, we can distinguish two families: the 'pseudopotential' methods and the 'model potential' methods. The pseudopotential methods are ultimately based on the Phillips-Kleinman equation [192] and handle valence nodeless pseudo orbitals. The model potential methods are based on the Huzinaga equation [193] and handle node-showing valence orbitals; the AIMP method belongs to this family. Here, when we use the general term ECP we will be referring to the more particular of AIMP. According to its characteristics, the AIMP method can be also applied to represent frozen-ions in ionic lattices in embedded cluster calculations; in this case, we will not be very strict in the nomenclature and we will also call ECP's to the frozen-ion (embedding) *ab initio* model potentials.

The effective potentials in the libraries include the effects of the atomic core wave functions (embedding ion wave functions) through the following operators:

- a local representation of the core (ion) Coulomb operator,
- a non-local spectral representation of the core (ion) exchange operator,
- a core (ion) projection operator,
- a spectral representation of the relativistic mass-velocity and Darwin operators corresponding to the valence orbitals, if the Cowan-Griffin-based scalar relativistic CG-AIMP method [187] is used.
- a spectral representation of the relativistic no-pair Douglas-Kroll operators, if the scalar relativistic no-pair Douglas-Kroll NP-AIMP method [188, 189, 190] is used.

Given the quality and non-parametric nature of the operators listed above, the flexibility of the basis sets to be used with the AIMP's is crucial, as in any *ab initio* method.

The valence basis sets included in the libraries have been obtained by energy minimization in atomic valence-electron calculations, following standard optimization procedures. All the experience gathered in the design of molecular basis sets starting from all-electron atomic basis sets, and in particular from segmented minimal ones, is directly applicable to the AIMP valence basis sets included in the libraries. They are, for non-relativistic and relativistic Cowan-Griffin AIMPs, minimal basis sets with added functions, such as polarization and diffuse functions; in consequence, the minimal sets should be split in molecular calculations in order to get reasonable sets (a splitting pattern is recommended in the library for every set); the splitting can be done by means of 'the basis set label'. For the relativistic no-pair Douglas-Kroll AIMPs contracted valence basis sets are given directly in a form which is recommended in molecular calculations, i.e. they are of triple zeta quality in the outer shells and contain polarization functions. In both cases these *valence* basis sets contain very *inner* primitive GTF's: They are necessary since, typical to a model potential method, the valence orbitals will show correct nodal structure. Finally, it must be noted that the core AIMP's can be safely mixed together with all-electron basis sets.

In AIMP *embedded cluster calculations*, the cluster basis set, which must be decided upon by the user, should be designed following high quality standard procedures. Very rigid cluster basis sets should not be used. In particular, the presence of the necessary embedding projection operators, which prevent the cluster densities from collapsing onto the crystal lattice, demands flexible cluster bases, including, eventually, components outside the cluster volume.[194] The use of flexible cluster basis sets is then a necessary requirement to avoid artificial frontier effects, not ascribable to the AIMP embedding potentials. This requirement is unavoidable, anyway, if good correlated wave functions are to be calculated for the cluster. Finally, one must remember that the AIMP method does exclude any correlation between the cluster electronic group and the embedding crystal components; in other words, only intra-cluster correlation effects can be accounted for in AIMP embedded cluster calculations. Therefore the cluster-environment partition and the choice of the cluster wave function must be done accordingly. In particular, the use of one-atom clusters is not recommended.

Core- and embedding- AIMP's can be combined in a natural way in valence-electron, embedded cluster calculations. They can be used with any of the different types of wave functions that can be calculated with MOLCAS.

Core AIMP's The list of core potentials and valence basis sets available in the *ECP* library follows. Although AIMP's exist in the literature for different core sizes, this library includes only those recommended by the authors after numerical experimentation. Relativistic CG-AIMP's and NP-AIMP's, respectively, and nonrelativistic NR-AIMP's are included. Each entry of the CG-AIMP's and the NR-AIMP's in the list is accompanied with a recommended contraction pattern (to be used in the fifth field). The NP-AIMP basis sets are given explicitly in the recommended contraction pattern. For the third-row transition metals two NP-AIMP basis sets are provided which differ in the number of primitive and contracted f GTFs. For further details, please refer to the literature.[190] For more information about a particular entry consult the *ECP* library.

```
1.  Cowan-Griffin-relativistic core AIMP's:  CG-AIMP

  1.1.  Main Group Elements

    1.1.1.  Alkaline Elements
      /Li.ECP.Barandiaran.5s1p.1s1p.1e-CG-AIMP.                2s1p
      /Na.ECP.Barandiaran.7s6p.1s2p.7e-CG-AIMP.                2s3p
      /K.ECP.Barandiaran.9s7p.1s2p.7e-CG-AIMP.                 2s3p
```

```
      /Rb.ECP.Barandiaran.11s9p6d.1s2p1d.7e-CG-AIMP.            2s3p1d
      /Cs.ECP.Barandiaran.13s11p8d.1s2p1d.7e-CG-AIMP.           3s3p1d
    1.1.2.  Alkaline Earth Elements
      /Be.ECP.Barandiaran.5s1p.1s1p.2e-CG-AIMP.                 2s1p
      /Mg.ECP.Barandiaran.7s6p1d.1s2p1d.8e-CG-AIMP.             2s3p1d
      /Ca.ECP.Barandiaran.9s7p5d.1s2p3d.8e-CG-AIMP.             2s3p3d
      /Sr.ECP.Barandiaran.11s9p6d.1s2p1d.8e-CG-AIMP.            3s3p3d
      /Ba.ECP.Barandiaran.13s11p8d.1s2p1d.8e-CG-AIMP.           3s3p3d
    1.1.3.  Group IIIA Elements
      /B.ECP.Barandiaran.5s5p1d.1s1p1d.3e-CG-AIMP.              2s3p1d
      /Al.ECP.Barandiaran.7s6p1d.1s1p1d.3e-CG-AIMP.             2s3p1d
      /Ga.ECP.Barandiaran.9s8p4d.1s1p2d.3e-CG-AIMP.             3s3p2d
      /In.ECP.Barandiaran.11s10p7d.1s1p2d.13e-CG-AIMP.          3s3p3d
      /Tl.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.13e-CG-AIMP.      3s4p3d2f
    1.1.4.  Group IVA Elements
      /C.ECP.Barandiaran.5s5p1d.1s1p1d.4e-CG-AIMP.              2s3p1d
      /Si.ECP.Barandiaran.7s6p1d.1s1p1d.4e-CG-AIMP.             2s3p1d
      /Ge.ECP.Barandiaran.9s8p4d.1s1p2d.4e-CG-AIMP.             3s3p2d
      /Sn.ECP.Barandiaran.11s10p7d.1s1p2d.14e-CG-AIMP.          3s3p3d
      /Pb.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.14e-CG-AIMP.      3s4p3d2f
    1.1.5.  Group VA Elements
      /N.ECP.Barandiaran.5s5p1d.1s1p1d.5e-CG-AIMP.              2s3p1d
      /P.ECP.Barandiaran.7s6p1d.1s1p1d.5e-CG-AIMP.              2s3p1d
      /As.ECP.Barandiaran.9s8p4d.1s1p2d.5e-CG-AIMP.             3s3p2d
      /Sb.ECP.Barandiaran.11s10p7d.1s1p2d.15e-CG-AIMP.          3s3p3d
      /Bi.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.15e-CG-AIMP.      3s4p3d2f
    1.1.6.  Group VIA Elements (Calcogens)
      /O.ECP.Barandiaran.5s6p1d.1s2p1d.6e-CG-AIMP.              2s4p1d
      /S.ECP.Barandiaran.7s6p1d.1s1p1d.6e-CG-AIMP.              2s3p1d
      /Se.ECP.Barandiaran.9s8p4d.1s1p2d.6e-CG-AIMP.             3s3p2d
      /Te.ECP.Barandiaran.11s10p7d.1s1p2d.16e-CG-AIMP.          3s3p3d
      /Po.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.16e-CG-AIMP.      3s4p3d2f
    1.1.7.  Group VIIA Elements (Halogens)
      /F.ECP.Barandiaran.5s6p1d.1s2p1d.7e-CG-AIMP.              2s4p1d
      /Cl.ECP.Barandiaran.7s7p1d.1s2p1d.7e-CG-AIMP.             2s4p1d
      /Br.ECP.Barandiaran.9s8p4d.1s1p2d.7e-CG-AIMP.             3s4p2d
      /I.ECP.Barandiaran.11s10p7d.1s1p2d.17e-CG-AIMP.           3s4p3d
      /At.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.17e-CG-AIMP.      3s4p3d2f
    1.1.8.  Group 0 Elements (Noble Gases)
      /Ne.ECP.Barandiaran.5s5p1d.1s1p1d.8e-CG-AIMP.             2s3p1d
      /Ar.ECP.Barandiaran.7s6p1d.1s1p1d.8e-CG-AIMP.             2s3p1d
      /Kr.ECP.Barandiaran.9s8p4d.1s1p2d.8e-CG-AIMP.             3s3p2d
      /Xe.ECP.Barandiaran.11s10p7d.1s1p2d.18e-CG-AIMP.          3s3p3d
      /Rn.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.18e-CG-AIMP.      3s4p3d2f

  1.2.  Transition Metal Elements

    1.2.1.  First Series Transition Metal Elements
      /Sc.ECP.Barandiaran.9s6p6d.1s2p2d.9e-CG-AIMP.             3s3p4d
```

```
    /Ti.ECP.Barandiaran.9s6p6d.1s2p2d.10e-CG-AIMP.            3s3p4d
    /V.ECP.Barandiaran.9s6p6d.1s2p2d.11e-CG-AIMP.             3s3p4d
    /Cr.ECP.Barandiaran.9s6p6d.1s2p2d.12e-CG-AIMP.            3s3p4d
    /Mn.ECP.Barandiaran.9s6p6d.1s2p2d.13e-CG-AIMP.            3s3p4d
    /Fe.ECP.Barandiaran.9s6p6d.1s2p2d.14e-CG-AIMP.            3s3p4d
    /Co.ECP.Barandiaran.9s6p6d.1s2p2d.15e-CG-AIMP.            3s3p4d
    /Ni.ECP.Barandiaran.9s6p6d.1s2p2d.16e-CG-AIMP.            3s3p4d
    /Cu.ECP.Barandiaran.9s6p6d.1s2p2d.17e-CG-AIMP.            3s3p4d
    /Zn.ECP.Barandiaran.9s6p5d.1s2p1d.18e-CG-AIMP.            3s3p3d
  1.2.2.  Second Series Transition Metal Elements
    /Y.ECP.Barandiaran.11s8p7d.1s2p2d.9e-CG-AIMP.             3s3p4d
    /Zr.ECP.Barandiaran.11s8p7d.1s2p2d.10e-CG-AIMP.           3s3p4d
    /Nb.ECP.Barandiaran.11s8p7d.1s2p2d.11e-CG-AIMP.           3s3p4d
    /Mo.ECP.Barandiaran.11s8p7d.1s2p2d.12e-CG-AIMP.           3s3p4d
    /Tc.ECP.Barandiaran.11s8p7d.1s2p2d.13e-CG-AIMP.           3s3p4d
    /Ru.ECP.Barandiaran.11s8p7d.1s2p2d.14e-CG-AIMP.           3s3p4d
    /Rh.ECP.Barandiaran.11s8p7d.1s2p2d.15e-CG-AIMP.           3s3p4d
    /Pd.ECP.Barandiaran.11s8p7d.1s2p2d.16e-CG-AIMP.           3s3p4d
    /Ag.ECP.Barandiaran.11s8p7d.1s2p2d.17e-CG-AIMP.           3s3p4d
    /Cd.ECP.Barandiaran.11s8p6d.1s2p1d.18e-CG-AIMP.           3s3p3d
  1.2.3 Lanthanum and Third Series Transition Metal Elements
    /La.ECP.Casarrubios.13s10p8d.1s2p2d.9e-CG-AIMP.           3s3p3d
    /Hf.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.10e-CG-AIMP.      3s3p4d2f
    /Ta.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.11e-CG-AIMP.      3s3p4d2f
    /W.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.12e-CG-AIMP.       3s3p4d2f
    /Re.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.13e-CG-AIMP.      3s3p4d2f
    /Os.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.14e-CG-AIMP.      3s3p4d2f
    /Ir.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.15e-CG-AIMP.      3s3p4d2f
    /Pt.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.16e-CG-AIMP.      3s3p4d2f
    /Pt.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.16e-CG-AIMP-ave.  3s3p4d2f
    /Au.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.17e-CG-AIMP.      3s3p4d2f
    /Hg.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.18e-CG-AIMP.      3s3p4d2f

1.3.  Lanthanide Elements

    /Ce.ECP.Diaz-Megias.14s10p9d8f.2s1p1d1f.12e-CG-AIMP.      6s5p5d4f
    /Pr.ECP.Seijo.14s10p9d8f.2s1p1d1f.13e-CG-AIMP.            6s5p5d4f
    /Nd.ECP.Seijo.14s10p9d8f.2s1p1d1f.14e-CG-AIMP.            6s5p5d4f
    /Pm.ECP.Seijo.14s10p9d8f.2s1p1d1f.15e-CG-AIMP.            6s5p5d4f
    /Sm.ECP.Seijo.14s10p9d8f.2s1p1d1f.16e-CG-AIMP.            6s5p5d4f
    /Eu.ECP.Seijo.14s10p9d8f.2s1p1d1f.17e-CG-AIMP.            6s5p5d4f
    /Gd.ECP.Seijo.14s10p9d8f.2s1p1d1f.18e-CG-AIMP.            6s5p5d4f
    /Tb.ECP.Seijo.14s10p9d8f.2s1p1d1f.19e-CG-AIMP.            6s5p5d4f
    /Dy.ECP.Seijo.14s10p9d8f.2s1p1d1f.20e-CG-AIMP.            6s5p5d4f
    /Ho.ECP.Seijo.14s10p9d8f.2s1p1d1f.21e-CG-AIMP.            6s5p5d4f
    /Er.ECP.Seijo.14s10p9d8f.2s1p1d1f.22e-CG-AIMP.            6s5p5d4f
    /Tm.ECP.Seijo.14s10p9d8f.2s1p1d1f.23e-CG-AIMP.            6s5p5d4f
    /Yb.ECP.Seijo.14s10p9d8f.2s1p1d1f.24e-CG-AIMP.            6s5p5d4f
    /Lu.ECP.Seijo.14s10p9d8f.2s1p1d1f.25e-CG-AIMP.            6s5p5d4f
```

1.4. Actinide Elements

/Th.ECP.Seijo.14s10p11d9f.2s1p1d1f.12e-CG-AIMP.	6s5p5d4f
/Pa.ECP.Seijo.14s10p11d9f.2s1p1d1f.13e-CG-AIMP.	6s5p5d4f
/U.ECP.Seijo.14s10p11d9f.2s1p1d1f.14e-CG-AIMP.	6s5p5d4f
/Np.ECP.Seijo.14s10p11d9f.2s1p1d1f.15e-CG-AIMP.	6s5p5d4f
/Pu.ECP.Seijo.14s10p11d9f.2s1p1d1f.16e-CG-AIMP.	6s5p5d4f
/Am.ECP.Seijo.14s10p11d9f.2s1p1d1f.17e-CG-AIMP.	6s5p5d4f
/Cm.ECP.Seijo.14s10p11d9f.2s1p1d1f.18e-CG-AIMP.	6s5p5d4f
/Bk.ECP.Seijo.14s10p11d9f.2s1p1d1f.19e-CG-AIMP.	6s5p5d4f
/Cf.ECP.Seijo.14s10p11d9f.2s1p1d1f.20e-CG-AIMP.	6s5p5d4f
/Es.ECP.Seijo.14s10p11d9f.2s1p1d1f.21e-CG-AIMP.	6s5p5d4f
/Fm.ECP.Seijo.14s10p11d9f.2s1p1d1f.22e-CG-AIMP.	6s5p5d4f
/Md.ECP.Seijo.14s10p11d9f.2s1p1d1f.23e-CG-AIMP.	6s5p5d4f
/No.ECP.Seijo.14s10p11d9f.2s1p1d1f.24e-CG-AIMP.	6s5p5d4f
/Lr.ECP.Seijo.14s10p11d9f.2s1p1d1f.25e-CG-AIMP.	6s5p5d4f

2. Non-relativistic core AIMP's: NR-AIMP

2.1. Main Group Elements

2.1.1. Alkaline Elements

/Li.ECP.Huzinaga.5s1p.1s1p.1e-NR-AIMP.	2s1p
/Na.ECP.Seijo.7s6p.1s2p.7e-NR-AIMP.	2s3p
/K.ECP.Seijo.9s7p.1s2p.7e-NR-AIMP.	2s3p
/Rb.ECP.Seijo.11s9p6d.1s2p1d.7e-NR-AIMP.	2s3p1d
/Cs.ECP.Seijo.13s11p8d.1s2p1d.7e-NR-AIMP.	3s3p1d

2.1.2. Alkaline Earth Elements

/Be.ECP.Huzinaga.5s1p.1s1p.2e-NR-AIMP.	2s1p
/Mg.ECP.Seijo.7s6p1d.1s2p1d.8e-NR-AIMP.	2s3p1d
/Ca.ECP.Seijo.9s7p5d.1s2p3d.8e-NR-AIMP.	2s3p3d
/Sr.ECP.Seijo.11s9p7d.1s2p1d.8e-NR-AIMP.	3s3p3d
/Ba.ECP.Seijo.13s11p8d.1s2p1d.8e-NR-AIMP.	3s3p3d

2.1.3. Group IIIA Elements

/B.ECP.Huzinaga.5s5p1d.1s1p1d.3e-NR-AIMP.	2s3p1d
/Al.ECP.Huzinaga.7s6p1d.1s1p1d.3e-NR-AIMP.	2s3p1d
/Ga.ECP.Huzinaga.9s8p4d.1s1p2d.3e-NR-AIMP.	3s3p2d
/In.ECP.Barandiaran.11s10p7d.1s1p2d.13e-NR-AIMP.	3s3p3d
/Tl.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.13e-NR-AIMP.	3s4p3d2f

2.1.4. Group IVA Elements

/C.ECP.Huzinaga.5s5p1d.1s1p1d.4e-NR-AIMP.	2s3p1d
/Si.ECP.Huzinaga.7s6p1d.1s1p1d.4e-NR-AIMP.	2s3p1d
/Ge.ECP.Huzinaga.9s8p4d.1s1p2d.4e-NR-AIMP.	3s3p2d
/Sn.ECP.Barandiaran.11s10p7d.1s1p2d.14e-NR-AIMP.	3s3p3d
/Pb.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.14e-NR-AIMP.	3s4p3d2f

2.1.5. Group VA Elements

/N.ECP.Huzinaga.5s5p1d.1s1p1d.5e-NR-AIMP.	2s3p1d

```
    /P.ECP.Huzinaga.7s6p1d.1s1p1d.5e-NR-AIMP.                 2s3p1d
    /As.ECP.Huzinaga.9s8p4d.1s1p2d.5e-NR-AIMP.                3s3p2d
    /Sb.ECP.Barandiaran.11s10p7d.1s1p2d.15e-NR-AIMP.          3s3p3d
    /Bi.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.15e-NR-AIMP.      3s4p3d2f
  2.1.6.  Group VIA Elements (Calcogens)
    /O.ECP.Huzinaga.5s6p1d.1s2p1d.6e-NR-AIMP.                 2s4p1d
    /S.ECP.Huzinaga.7s6p1d.1s1p1d.6e-NR-AIMP.                 2s3p1d
    /Se.ECP.Huzinaga.9s8p4d.1s1p2d.6e-NR-AIMP.                3s3p2d
    /Te.ECP.Barandiaran.11s10p7d.1s1p2d.16e-NR-AIMP.          3s3p3d
    /Po.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.16e-NR-AIMP.      3s4p3d2f
  2.1.7.  Group VIIA Elements (Halogens)
    /F.ECP.Huzinaga.5s6p1d.1s2p1d.7e-NR-AIMP.                 2s4p1d
    /Cl.ECP.Huzinaga.7s7p1d.1s2p1d.7e-NR-AIMP.                2s4p1d
    /Br.ECP.Huzinaga.9s8p4d.1s1p2d.7e-NR-AIMP.                3s4p2d
    /I.ECP.Barandiaran.11s10p7d.1s1p2d.17e-NR-AIMP.           3s4p3d
    /At.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.17e-NR-AIMP.      3s4p3d2f
  2.1.8.  Group 0 Elements (Noble Gases)
    /Ne.ECP.Huzinaga.5s5p1d.1s1p1d.8e-NR-AIMP.                2s3p1d
    /Ar.ECP.Huzinaga.7s6p1d.1s1p1d.8e-NR-AIMP.                2s3p1d
    /Kr.ECP.Huzinaga.9s8p4d.1s1p2d.8e-NR-AIMP.                3s3p2d
    /Xe.ECP.Barandiaran.11s10p7d.1s1p2d.18e-NR-AIMP.          3s3p3d
    /Rn.ECP.Barandiaran.13s12p8d5f.1s1p2d1f.18e-NR-AIMP.      3s4p3d2f

2.2.  Transition Metal Elements

  2.2.1.  First Series Transition Metal Elements
    /Sc.ECP.Seijo.9s6p6d.1s2p2d.9e-NR-AIMP.                   3s3p4d
    /Ti.ECP.Seijo.9s6p6d.1s2p2d.10e-NR-AIMP.                  3s3p4d
    /V.ECP.Seijo.9s6p6d.1s2p2d.11e-NR-AIMP.                   3s3p4d
    /Cr.ECP.Seijo.9s6p6d.1s2p2d.12e-NR-AIMP.                  3s3p4d
    /Mn.ECP.Seijo.9s6p6d.1s2p2d.13e-NR-AIMP.                  3s3p4d
    /Fe.ECP.Seijo.9s6p6d.1s2p2d.14e-NR-AIMP.                  3s3p4d
    /Co.ECP.Seijo.9s6p6d.1s2p2d.15e-NR-AIMP.                  3s3p4d
    /Ni.ECP.Seijo.9s6p6d.1s2p2d.16e-NR-AIMP.                  3s3p4d
    /Cu.ECP.Seijo.9s6p6d.1s2p2d.17e-NR-AIMP.                  3s3p4d
    /Zn.ECP.Seijo.9s6p5d.1s2p1d.18e-NR-AIMP.                  3s3p3d
  2.2.2.  Second Series Transition Metal Elements
    /Y.ECP.Barandiaran.11s8p7d.1s2p2d.9e-NR-AIMP.             3s3p4d
    /Zr.ECP.Barandiaran.11s8p7d.1s2p2d.10e-NR-AIMP.           3s3p4d
    /Nb.ECP.Barandiaran.11s8p7d.1s2p2d.11e-NR-AIMP.           3s3p4d
    /Mo.ECP.Barandiaran.11s8p7d.1s2p2d.12e-NR-AIMP.           3s3p4d
    /Tc.ECP.Barandiaran.11s8p7d.1s2p2d.13e-NR-AIMP.           3s3p4d
    /Ru.ECP.Barandiaran.11s8p7d.1s2p2d.14e-NR-AIMP.           3s3p4d
    /Rh.ECP.Barandiaran.11s8p7d.1s2p2d.15e-NR-AIMP.           3s3p4d
    /Pd.ECP.Barandiaran.11s8p7d.1s2p2d.16e-NR-AIMP.           3s3p4d
    /Ag.ECP.Barandiaran.11s8p7d.1s2p2d.17e-NR-AIMP.           3s3p4d
    /Cd.ECP.Barandiaran.11s8p6d.1s2p1d.18e-NR-AIMP.           3s3p3d
  2.2.3 Lanthanum and Third Series Transition Metal Elements
    /La.ECP.Casarrubios.13s10p8d.1s2p2d.9e-NR-AIMP.           3s3p3d
```

/Hf.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.10e-NR-AIMP.	3s3p4d2f
/Ta.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.11e-NR-AIMP.	3s3p4d2f
/W.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.12e-NR-AIMP.	3s3p4d2f
/Re.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.13e-NR-AIMP.	3s3p4d2f
/Os.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.14e-NR-AIMP.	3s3p4d2f
/Ir.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.15e-NR-AIMP.	3s3p4d2f
/Pt.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.16e-NR-AIMP.	3s3p4d2f
/Au.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.17e-NR-AIMP.	3s3p4d2f
/Hg.ECP.Casarrubios.13s10p9d5f.1s2p2d1f.18e-NR-AIMP.	3s3p4d2f

3. Relativistic no-pair Douglas-Kroll core AIMP's: NP-AIMP

3.1. Transition Metal Elements

3.1.1. First Series Transition Metal Elements

/Sc.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.11e-NP-AIMP.
/Ti.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.12e-NP-AIMP.
/V.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.13e-NP-AIMP.
/Cr.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.14e-NP-AIMP.
/Mn.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.15e-NP-AIMP.
/Fe.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.16e-NP-AIMP.
/Co.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.17e-NP-AIMP.
/Ni.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.18e-NP-AIMP.
/Cu.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.19e-NP-AIMP.
/Zn.ECP.Rakowitz.9s6p6d3f.5s4p4d1f.20e-NP-AIMP.

3.1.2. Second Series Transition Metal Elements

/Y.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.11e-NP-AIMP.
/Zr.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.12e-NP-AIMP.
/Nb.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.13e-NP-AIMP.
/Mo.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.14e-NP-AIMP.
/Tc.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.15e-NP-AIMP.
/Ru.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.16e-NP-AIMP.
/Rh.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.17e-NP-AIMP.
/Pd.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.18e-NP-AIMP.
/Ag.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.19e-NP-AIMP.
/Cd.ECP.Rakowitz.11s8p7d3f.5s4p4d1f.20e-NP-AIMP.

3.1.3. Third Series Transition Metal Elements

/Hf.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.12e-NP-AIMP.
/Hf.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.12e-NP-AIMP.
/Ta.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.13e-NP-AIMP.
/Ta.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.13e-NP-AIMP.
/W.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.14e-NP-AIMP.
/W.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.14e-NP-AIMP.
/Re.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.15e-NP-AIMP.
/Re.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.15e-NP-AIMP.
/Os.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.16e-NP-AIMP.
/Os.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.16e-NP-AIMP.
/Ir.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.17e-NP-AIMP.

```
/Ir.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.17e-NP-AIMP.
/Pt.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.18e-NP-AIMP.
/Pt.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.18e-NP-AIMP.
/Au.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.19e-NP-AIMP.
/Au.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.19e-NP-AIMP.
/Hg.ECP.Rakowitz.13s10p9d6f.5s4p4d2f.20e-NP-AIMP.
/Hg.ECP.Rakowitz.13s10p9d1f.5s4p4d1f.20e-NP-AIMP.
```

Embedding AIMP's The list of complete-ion embedding potentials available in the library *ECP* follows.

1. Elpasolites

   1.1. K_2NaGaF_6(a = 8.246 Å; 0 Kbar)
   /K.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6.
   /Na.ECP.Barandiaran.6s3p.1s1p.0e-AIMP-K2NaGaF6.
   /Na.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6.
   /Ga.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6.
   /F.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6.
   1.2. K_2NaGaF_6(a = 8.000 Å; 60 Kbar)
   /K.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6-60Kbar.
   /Na.ECP.Barandiaran.6s3p.1s1p.0e-AIMP-K2NaGaF6-60Kbar.
   /Na.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6-60Kbar.
   /Ga.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6-60Kbar.
   /F.ECP.Barandiaran.0s.0s.0e-AIMP-K2NaGaF6-60Kbar.
   1.3. Cs_2NaYCl_6(a = 10.7396 Å, x=0.2439)
   /Cs.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYCl6.
   /Na.ECP.Abdalla.7s4p.1s1p.0e-AIMP-Cs2NaYCl6.
   /Na.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYCl6.
   /Y.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYCl6.
   /Cl.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYCl6.
   1.4. Cs_2NaYBr_6(a = 11.3047 Å, x=0.2446)
   /Cs.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYBr6.
   /Na.ECP.Abdalla.14s7p.1s2p.0e-AIMP-Cs2NaYBr6.
   /Na.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYBr6.
   /Y.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYBr6.
   /Br.ECP.Abdalla.0s.0s.0e-AIMP-Cs2NaYBr6.

2. Fluoro-Perovskites

   2.1. $KMgF_3$(a = 3.973 Å)
   /K.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.
   /Mg.ECP.Lopez-Moraza.7s4p.1s1p.0e-AIMP-KMgF3.
   /Mg.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.
   /F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KMgF3.
   2.2. $KZnF_3$(a = 4.054 Å)
   /K.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KZnF3.

/Zn.ECP.Lopez-Moraza.11s8p5d.1s1p1d.0e-AIMP-KZnF3.
/Zn.ECP.Lopez-Moraza.11s8p.1s1p.0e-AIMP-KZnF3.
/Zn.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KZnF3.
/F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KZnF3.

2.3. $KCdF_3$(a = 4.302 Å)
/K.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KCdF3.
/Cd.ECP.Lopez-Moraza.11s7p6d.1s1p1d.0e-AIMP-KCdF3.
/Cd.ECP.Lopez-Moraza.11s7p.1s1p.0e-AIMP-KCdF3.
/Cd.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KCdF3.
/F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-KCdF3.

2.4. $CsCaF_3$(a = 4.526 Å)
/Cs.ECP.Lopez-Moraza.0s.0s.0e-AIMP-CsCaF3.
/Ca.ECP.Lopez-Moraza.10s7p.1s1p.0e-AIMP-CsCaF3.
/Ca.ECP.Lopez-Moraza.0s.0s.0e-AIMP-CsCaF3.
/F.ECP.Lopez-Moraza.0s.0s.0e-AIMP-CsCaF3.

3. Rock salt structure oxides and halides

3.1. MgO (a = 4.2112 Å)
/Mg.ECP.Pascual.10s4p.1s1p.0e-AIMP-MgO.
/Mg.ECP.Pascual.0s.0s.0e-AIMP-MgO.
/Mg.ECP.Pascual.0s.0s.0e-AIMP-MgO-0.65.
/O.ECP.Pascual.8s6p.1s1p.0e-AIMP-MgO.
/O.ECP.Pascual.0s.0s.0e-AIMP-MgO.
/O.ECP.Pascual.0s.0s.0e-AIMP-MgO-0.65.

3.2. CaO (a = 4.8105 Å)
/Ca.ECP.Pascual.13s7p.1s1p.0e-AIMP-CaO.
/Ca.ECP.Pascual.0s.0s.0e-AIMP-CaO.
/O.ECP.Pascual.0s.0s.0e-AIMP-CaO.

3.3. SrO (a = 5.1602 Å)
/Sr.ECP.Pascual.0s.0s.0e-AIMP-SrO.
/O.ECP.Pascual.0s.0s.0e-AIMP-SrO.
/Sr.ECP.Pascual.0s.0s.0e-AIMP-SrO-QR.
/O.ECP.Pascual.0s.0s.0e-AIMP-SrO-QR.

3.4. NiO (a = 4.164 Å)
/Ni.ECP.Seijo.0s.0s.0e-AIMP-NiO.
/O.ECP.Seijo.0s.0s.0e-AIMP-NiO.

3.5. LiF (a = 4.028 Å)
/Li.ECP.Seijo.4s.1s.0e-AIMP-LiF.
/Li.ECP.Seijo.0s.0s.0e-AIMP-LiF.
/F.ECP.Seijo.0s.0s.0e-AIMP-LiF.

3.6. NaF (a = 4.634 Å)
/Na.ECP.Seijo.6s3p.1s1p.0e-AIMP-NaF.
/Na.ECP.Seijo.0s.0s.0e-AIMP-NaF.
/F.ECP.Seijo.0s.0s.0e-AIMP-NaF.

3.7. KF (a = 5.348 Å)
/K.ECP.Seijo.10s7p.1s1p.0e-AIMP-KF.
/K.ECP.Seijo.0s.0s.0e-AIMP-KF.
/F.ECP.Seijo.0s.0s.0e-AIMP-KF.

3.8. NaCl (a = 5.64056 Å)
`/Na.ECP.Seijo.6s3p.1s1p.0e-AIMP-NaCl.`
`/Na.ECP.Seijo.0s.0s.0e-AIMP-NaCl.`
`/Cl.ECP.Seijo.0s.0s.0e-AIMP-NaCl.`

4. Fluorites

4.1. CaF_2(a = 5.46294 Å)
`/Ca.ECP.Pascual.0s.0s.0e-AIMP-CaF2.`
`/F.ECP.Pascual.0s.0s.0e-AIMP-CaF2.`
4.2. SrF_2
`/Sr.ECP.Pascual.0s.0s.0e-AIMP-SrF2.`
`/F.ECP.Pascual.0s.0s.0e-AIMP-SrF2.`
4.3. BaF_2
`/Ba.ECP.Pascual.0s.0s.0e-AIMP-BaF2.`
`/F.ECP.Pascual.0s.0s.0e-AIMP-BaF2.`
4.4. CdF_2
`/Cd.ECP.Pascual.0s.0s.0e-AIMP-CdF2.`
`/F.ECP.Pascual.0s.0s.0e-AIMP-CdF2.`

5. Miscellany

5.1. Cs_2GeF_6
`/Cs.ECP.Casarrubios.0s.0s.0e-AIMP-Cs2GeF6.`
`/Ge.ECP.Casarrubios.0s.0s.0e-AIMP-Cs2GeF6.`
`/F.ECP.Casarrubios.0s.0s.0e-AIMP-Cs2GeF6.`

The ECP libraries have also been extended to include the so-called nodeless ECPs or pseudo potentials based on the Phillips-Kleinman equation [192]. These are included both as explicit and implicit operators. Following the work by M. Pelissier and co-workers [195] the operators of nodeless ECPs can implicitly be fully expressed via spectral representation of operators. The explicit libraries are the ECP.STOLL and ECP.HAY-WADT files, all other files are for the implicitly expressed operator. In the list of nodeless ECPs the Hay and Wadt's family of ECPs (LANL2DZ ECPs) [196, 197, 198] has been included in addition to the popular set of the so-called Stoll and Dolg ECPs [199, 200, 201, 202, 203, 204, 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221, 222, 223]. Both of them in either the explicit form labeled as HAY-WADT and STOLL, or in the implicit form labeled as HW and DOLG. The latter include the recently developed ANO-basis sets for actinides [223]

1. Hay and Wadt (LANL2DZ) ECP's

`/Na.ECP.HW.3s3p.2s2p.1e-LANL2DZ.`
`/Mg.ECP.HW.3s3p.2s2p.2e-LANL2DZ.`
`/Al.ECP.HW.3s3p.2s2p.3e-LANL2DZ.`
`/Si.ECP.HW.3s3p.2s2p.4e-LANL2DZ.`
`/P.ECP.HW.3s3p.2s2p.5e-LANL2DZ.`
`/S.ECP.HW.3s3p.2s2p.6e-LANL2DZ.`
`/Cl.ECP.HW.3s3p.2s2p.7e-LANL2DZ.`

```
/Ar.ECP.HW.3s3p.2s2p.8e-LANL2DZ.
/K.ECP.HW.5s5p.3s3p.9e-LANL2DZ.
/Ca.ECP.HW.5s5p.3s3p.10e-LANLDZ.
/Sc.ECP.HW.5s5p5d.3s3p2d.11e-LANL2DZ.
/Ti.ECP.HW.5s5p5d.3s3p2d.12e-LANL2DZ.
/V.ECP.HW.5s5p5d.3s3p2d.13e-LANL2DZ.
/Cr.ECP.HW.5s5p5d.3s3p2d.14e-LANL2DZ.
/Mn.ECP.HW.5s5p5d.3s3p2d.15e-LANL2DZ.
/Fe.ECP.HW.5s5p5d.3s3p2d.16e-LANL2DZ.
/Co.ECP.HW.5s5p5d.3s3p2d.17e-LANL2DZ.
/Ni.ECP.HW.5s5p5d.3s3p2d.18e-LANL2DZ.
/Cu.ECP.HW.5s5p5d.3s3p2d.19e-LANL2DZ.
/Zn.ECP.HW.3s2p5d.2s2p2d.12e-LANL2DZ.
/Ga.ECP.HW.3s3p.2s2p.3e-LANL2DZ.
/Ge.ECP.HW.3s3p.2s2p.4e-LANL2DZ.
/As.ECP.HW.3s3p.2s2p.5e-LANL2DZ.
/Se.ECP.HW.3s3p.2s2p.6e-LANL2DZ.
/Br.ECP.HW.3s3p.2s2p.7e-LANL2DZ.
/Kr.ECP.HW.3s3p.2s2p.8e-LANL2DZ.
/Rb.ECP.HW.5s6p.3s3p.9e-LANLDZ.
/Sr.ECP.HW.5s6p.3s3p.10e-LANLDZ.
/Y.ECP.HW.5s6p4d.3s3p2d.11e-LANL2DZ.
/Zr.ECP.HW.5s6p4d.3s3p2d.12e-LANL2DZ.
/Nb.ECP.HW.5s6p4d.3s3p2d.13e-LANL2DZ.
/Mo.ECP.HW.5s6p4d.3s3p2d.14e-LANL2DZ.
/Tc.ECP.HW.5s6p4d.3s3p2d.15e-LANL2DZ.
/Ru.ECP.HW.5s6p4d.3s3p2d.16e-LANL2DZ.
/Rh.ECP.HW.5s6p4d.3s3p2d.17e-LANL2DZ.
/Pd.ECP.HW.5s6p4d.3s3p2d.18e-LANL2DZ.
/Ag.ECP.HW.5s6p4d.3s3p2d.19e-LANL2DZ.
/Cd.ECP.HW.3s3p4d.2s2p2d.12e-LANL2DZ.
/In.ECP.HW.3s3p.2s2p.3e-LANL2DZ.
/Sn.ECP.HW.3s3p.2s2p.4e-LANL2DZ.
/Sb.ECP.HW.3s3p.2s2p.5e-LANL2DZ.
/Te.ECP.HW.3s3p.2s2p.6e-LANL2DZ.
/I.ECP.HW.3s3p.2s2p.7e-LANL2DZ.
/Xe.ECP.HW.3s3p.2s2p.8e-LANL2DZ.
/Cs.ECP.HW.5s6p.3s3p.9e-LANL2DZ.
/Ba.ECP.HW.5s6p.3s3p.10e-LANL2DZ.
/La.ECP.HW.5s6p3d.3s3p2d.11e-LANL2DZ.
/Hf.ECP.HW.5s6p3d.3s3p2d.12e-LANL2DZ.
/Ta.ECP.HW.5s6p3d.3s3p2d.13e-LANL2DZ.
/W.ECP.HW.5s6p3d.3s3p2d.14e-LANL2DZ.
/Re.ECP.HW.5s6p3d.3s3p2d.15e-LANL2DZ.
/Os.ECP.HW.5s6p3d.3s3p2d.16e-LANL2DZ.
/Ir.ECP.HW.5s6p3d.3s3p2d.17e-LANL2DZ.
/Pt.ECP.HW.5s6p3d.3s3p2d.18e-LANL2DZ.
/Au.ECP.HW.5s6p3d.3s3p2d.19e-LANL2DZ.
/Hg.ECP.HW.3s3p3d.2s2p2d.12e-LANL2DZ.
```

```
    /Tl.ECP.HW.3s3p3d.2s2p2d.13e-LANL2DZ.
    /Pb.ECP.HW.3s3p.2s2p.4e-LANL2DZ.
    /Bi.ECP.HW.3s3p.2s2p.5e-LANL2DZ.

  2.  "Dolg" ECP's

    /Li.ECP.Dolg.4s4p.2s2p.2e-SDF.
    /Be.ECP.Dolg.4s4p.2s2p.2e-SDF.
    /B.ECP.Dolg.4s4p.2s2p.3e-MWB.
    /C.ECP.Dolg.4s4p.2s2p.4e-MWB.
    /N.ECP.Dolg.4s4p.2s2p.5e-MWB.
    /O.ECP.Dolg.4s5p.2s3p.6e-MWB.
    /F.ECP.Dolg.4s5p.2s3p.7e-MWB.
    /Ne.ECP.Dolg.7s7p3d1f.4s4p3d1f.8e-MWB.
    /Na.ECP.Dolg.4s4p.2s2p.1e-SDF.
    /Mg.ECP.Dolg.4s4p.2s2p.2e-SDF.
    /Al.ECP.Dolg.4s4p.2s2p.3e-MWB.
    /Si.ECP.Dolg.4s4p.2s2p.4e-MWB.
    /P.ECP.Dolg.4s4p.2s2p.5e-MWB.
    /S.ECP.HEULLY.4s5p2d.2s3p2d.6e.
    /Cl.ECP.Dolg.4s5p.2s3p.7e-MWB.
    /Ar.ECP.Dolg.6s6p3d1f.4s4p3d1f.8e-MWB.
    /K.ECP.Dolg.7s6p.5s4p.9e-MWB.
    /Ca.ECP.Dolg.6s6p5d.4s4p2d.10e-MWB.
    /Sc.ECP.Dolg.8s7p6d1f.6s5p3d1f.11e-MWB.
    /Ti.ECP.Dolg.8s7p6d1f.6s5p3d1f.12e-MDF.
    /V.ECP.Dolg.8s7p6d1f.6s5p3d1f.13e-MDF.
    /Cr.ECP.Dolg.8s7p6d.6s5p3d.14e-MDF.
    /Mn.ECP.Dolg.8s7p6d1f.6s5p3d1f.15e-MDF.
    /Fe.ECP.Dolg.8s7p6d1f.6s5p3d1f.16-MDF.
    /Co.ECP.Dolg.8s7p6d1f.6s5p3d1f.17e-MDF.
    /Ni.ECP.Dolg.8s7p6d1f.6s5p3d1f.18e-MDF.
    /Cu.ECP.Dolg.8s7p6d.6s5p3d.19e-MDF.
    /Zn.ECP.DOlg.8s7p6d.6s5p3d.20e-MDF.
    /Zn.ECP.Dolg.4s2p.3s2p.2e-MWB.
    /Ga.ECP.Dolg.4s4p.2s2p.3e-MWB.
    /Ge.ECP.Dolg.4s4p.2s2p.4e-MWB.
    /As.ECP.Dolg.4s4p.2s2p.5e-MWB.
    /Se.ECP.Dolg.4s5p.2s3p.6e-MWB.
    /Br.ECP.Dolg.6s6p1d.5s5p1d.7e-MWB.
    /Kr.ECP.Dolg.6s6p3d1f.4s4p3d1f.8e-MWB.
    /Rb.ECP.Dolg.7s6p.5s4p.9e-MWB.
    /Sr.ECP.Dolg.6s6p5d.4s4p2d.10e-MWB.
    /Y.ECP.Dolg.8s7p6d.6s5p3d.11e-MWB.
    /Zr.ECP.Dolg.8s7p6d.6s5p3d.12e-MWB.
    /Nb.ECP.Dolg.8s7p6d.6s5p3d.13e-MWB.
    /Mo.ECP.Dolg.8s7p6d.6s5p3d.14e-MWB.
    /Tc.ECP.Dolg.8s7p6d.6s5p3d.15e-MWB.
    /Ru.ECP.Dolg.8s7p6d.6s5p3d.16e-MWB.
```

```
/Rh.ECP.Dolg.8s7p6d.6s5p3d.17e-MWB.
/Pd.ECP.Dolg.8s7p6d.6s5p3d.18e-MWB.
/Ag.ECP.Dolg.8s7p6d.6s5p3d.19e-MWB.
/Cd.ECP.Dolg.8s7p6d.6s5p3d.20e-MWB.
/In.ECP.Dolg.4s4p.2s2p.3e-MWB.
/Sn.ECP.Dolg.4s4p.2s2p.4e-MWB.
/Sb.ECP.Dolg.4s4p.2s2p.5e-MWB.
/Te.ECP.Dolg.4s5p1d.2s3p1d.6e-MWB.
/I.ECP.Dolg.4s5p.2s3p.7e-MWB.
/Xe.ECP.Dolg.6s6p3d1f.4s4p3d1f.8e-MWB.
/Cs.ECP.Dolg.7s6p.5s4p.9e-MWB.
/Ba.ECP.Dolg.6s6p5d1f.4s4p2d.10e-MWB.
/La.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/La.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Ce.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Ce.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Pr.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Pr.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Nd.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Nd.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Pm.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Pm.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Sm.ECP.Dolg.12s11p9d8f6g.5s5p4d4f3g.34e-MWB.
/Sm.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Sm.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Eu.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Eu.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Gd.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Gd.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Tb.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Tb.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Dy.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Dy.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Ho.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Ho.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Er.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Er.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Tm.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Tm.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Yb.ECP.Dolg.7s6p5d.5s4p3d.10e-MWB.
/Lu.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Yb.ECP.Dolg.7s6p5d.5s4p3d.11e-MWB.
/Hf.ECP.Dolg.8s7p6d.6s5p3d.12e-MWB.
/Ta.ECP.Dolg.8s7p6d.6s5p3d.13e-MWB.
/W.ECP.Dolg.8s7p6d.6s5p3d.14e-MWB.
/Re.ECP.Dolg.8s7p6d.6s5p3d.15e-MWB.
/Os.ECP.Dolg.8s7p6d.6s5p3d.16e-MWB.
/Ir.ECP.Dolg.8s7p6d.6s5p3d.17e-MWB.
/Pt.ECP.Dolg.8s7p6d.6s5p3d.18e-MWB.
```

```
/Au.ECP.Dolg.8s7p6d.6s5p3d.19e-MWB.
/Hg.ECP.Dolg.8s7p6d.6s5p3d.20e-MWB.
/Hg.ECP.Dolg.4s4p1d.2s2p1d.2e-MWB.
/Tl.ECP.Dolg.4s4p1d.2s2p1d.3e-MWB.
/Pb.ECP.Dolg.4s4p1d.2s2p1d.4e-MWB.
/Bi.ECP.Dolg.4s4p1d.2s2p1d.5e-MWB.
/Po.ECP.Dolg.4s4p1d.2s2p1d.6e-MWB.
/At.ECP.Dolg.4s4p1d.2s2p1d.7e-MWB.
/Rn.ECP.Dolg.4s4p1d.2s2p1d.8e-MWB.
/Ac.ECP.Dolg.12s11p10d8f.8s7p6d4f.29e-MWB.
/Th.ECP.Dolg.12s11p7d8f.8s7p6d4f.30e-MWB.
/Pa.ECP.Dolg.12s11p10d8f.8s7p6d4f.31e-MWB.
/U.ECP.Dolg.12s11p10d8f.8s7p6d4f.32e-MWB.
/U.ECP.Dolg.12s11p10d8f2g.8s7p6d4f2g.32e-MWB.
/Np.ECP.Dolg.12s11p10d8f.8s7p6d4f.33e-MWB.
/Np.ECP.Dolg.12s11p10d8f2g.8s7p6d4f2g.33e-MWB.
/Pu.ECP.Dolg.12s11p10d8f.8s7p6d4f.34e-MWB.
/Pu.ECP.Dolg.12s11p10d8f2g.8s7p6d4f2g.34e-MWB.
/Am.ECP.Dolg.12s11p10d8f.8s7p6d4f.35e-MWB.
/Cm.ECP.Dolg.12s11p10d8f.8s7p6d4f.36e-MWB.
/Bk.ECP.Dolg.12s11p10d8f.8s7p6d4f.37e-MWB.
/Es.ECP.Dolg.12s11p10d8f.8s7p6d4f.39e-MWB.
/Cf.ECP.Dolg.12s11p10d8f.8s7p6d4f.38e-MWB.
/Fm.ECP.Dolg.12s11p10d8f.8s7p6d4f.40e-MWB.
/Ac.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.29e-MWB.
/Th.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.30e-MWB.
/Pa.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.31e-MWB.
/U.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.32e-MWB.
/Np.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.33e-MWB.
/Pu.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.34e-MWB.
/Am.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.35e-MWB.
/Cm.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.36e-MWB.
/Bk.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.37e-MWB.
/Cf.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.38e-MWB.
/Es.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.39e-MWB.
/Fm.ECP.Dolg.14s13p10d8f6g.6s6p5d4f3g.40e-MWB.
```

Structure of the ECP libraries

The start of a given basis set and AIMP is identified by the line

`/label`

where “label” is defined below, in the input description to `SEWARD`. Then, comment lines, effective charge, and basis set follow, with the same structure that the all-electron Basis Set Library (see items 1. to 4. in Sec. 8.36.) Next, the AIMP/ECP/PP is specified as follows:

5. The pseudo potentential approach [224, 225, 226], see eqs. (3) and (4) in Ref. [227], with the following lines:

   (a) The keyword PP On the same line follows the atomic symbol of the element, the number of core electrons (N_c) and L, where $L - 1$ is the largest angular momentum orbital belonging to the core. This line is followed by $L+1$ identical sections. The first of these sections is the so-called L potential and the subsequent sections corresponds to the S-L, P-L, D-L, etc. potentials. Each sections start with a line specifying the number of Gaussian terms in the potential. This line is then followed by a single line for each Gaussian specifying the powers (n_{kl}), the Gaussian exponent (ζ_{kl}), and the associated coefficient (d_{kl}).
   Note that the pseudo potential input is mutually exclusive to the M1, M2, COREREP, and PROJOP keywords!

6. The Coulomb local model potential, eq.(6) in Ref. [185] with the following lines:
   (a) The keyword M1, which identifies the terms with $n_k = 0$.
   (b) The number of terms. If greater than 0, lines 6c and 6d are read.
   (c) The exponents α_k.
   (d) The coefficients A_k (divided by the negative of the effective charge).
   (e) The keyword M2, which identifies the terms with $n_k = 1$.
   (f) The number of terms. If greater than 0, lines 6g and 6h are read.
   (g) The exponents α_k.
   (h) The coefficients A_k (divided by the negative of the effective charge).
7. A line with the keyword COREREP followed by another one with a real constant. This is not used now but it is reserved for future use.
8. The projection operator, eq.(3) in Ref. [185] with the following lines:
   (a) The keyword PROJOP.
   (b) The maximum angular momentum (l) of the frozen core (embedding) orbitals. Lines 8c to 8f are repeated for each angular momentum l.
   (c) The number of primitives and the number of orbitals (more properly, degenerate sets of orbitals or l-shells) for angular momentum l. As an option, these two integers can be followed by the occupation numbers of the l-shells; default values are 2 for l=0, 6 for l=1, etc.
   (d) The projection constants, $-2\varepsilon_c$.
   (e) The exponents of the primitive functions.
   (f) The coefficients of the orbitals, one per column, using general contraction format.
9. The spectral representation operator, eq.(7) in Ref. [185] for NR-AIMP, eq.(3) in Ref. [187] for relativistic CG-AIMP, and eqs.(1) and (7) in Ref. [190] for relativistic NP-AIMP, with the following lines:
   (a) The keyword SPECTRAL REPRESENTATION OPERATOR.
   (b) One of the keywords VALENCE, CORE, or EXTERNAL. VALENCE indicates that the set of primitive functions specified in the basis set data will be used for the spectral representation operator; this is the standard for ab initio *core* model potentials. CORE means that the set of primitives specified in the PROJOP section will be used instead; this is the standard for complete-ion ab initio *embedding* model

potentials. EXTERNAL means that a set of primitives specific for the spectral representation operator will be provided in the next lines. In this case the format is one line in which an integer number specifies the highest angular momentum of the external basis sets; then, for each angular momentum the input is formated as for lines 2, 3, and 4 in Sec. 8.36.

(c) The keyword EXCHANGE.

(d) For relativistic AIMPs one of the keywords NOPAIR or 1STORDER RELATIVISTIC CORRECTION. NOPAIR indicates that scalar relativistic no-pair Douglas-Kroll AIMP integrals are to be calculated. 1STORDER RELATIVISTIC CORRECTION means that Cowan-Griffin-based scalar relativistic AIMP, CG-AIMP's, are used. In the latter case, in the next line a *keyword* follows which, in the library ***QRPLIB***, identifies the starting of the numerical mass-velocity plus Darwin potentials (eq.(2) in Ref. [187]). (In ***QRPLIB*** a line with "*keyword* MV&DW POTENTIALS START" must exist, followed by the number of points in the radial logarithmic grid, the values of the radial coordinate r, and, for each valence orbital, its label (2S, 4P, etc), and the values of the mass-velocity plus Darwin potentials at the corresponding values of r; these data must end up with a line "*keyword* MV&DW POTENTIALS END".)

(e) The keyword END OF SPECTRAL REPRESENTATION OPERATOR.

Below is an example of an entry in the ***ECP*** library for an AIMP.

```
/S.ECP.Barandiaran.7s6p1d.1s1p1d.6e-CG-AIMP.       -- label (note that type is ECP)
Z.Barandiaran and L.Seijo, Can.J.Chem. 70(1992)409. -- 1st ref. line
core[Ne] val[3s,3p]  (61/411/1*)=2s3p1d recommended -- 2nd ref. line
*SQR-SP(7/6/1)                 (61/411/1)           -- comment line
  6.000000        2                                 -- eff. charge & highest ang.mom.
                                                    -- blank line
    7    1                                          -- 7s -> 1s
   1421.989530                                      -- s-exponent
   211.0266560                                      -- s-exponent
   46.72165060                                      -- s-exponent
   4.310564040                                      -- s-exponent
   1.966475840                                      -- s-exponent
   .4015383790                                      -- s-exponent
   .1453058790                                      -- s-exponent
   .004499703540                                    -- contr. coeff.
   .030157124800                                    -- contr. coeff.
   .089332590700                                    -- contr. coeff.
  -.288438151000                                    -- contr. coeff.
  -.279252515000                                    -- contr. coeff.
   .700286615000                                    -- contr. coeff.
   .482409523000                                    -- contr. coeff.
    6    1                                          -- 6p -> 1p
   78.08932440                                      -- p-exponent
   17.68304310                                      -- p-exponent
   4.966340810                                      -- p-exponent
   .5611646780                                      -- p-exponent
   .2130782690                                      -- p-exponent
   .8172415400E-01                                  -- p-exponent
  -.015853278200                                    -- contr. coeff.
```

```
  -.084808963800                                    -- contr. coeff.
  -.172934245000                                    -- contr. coeff.
   .420961662000                                    -- contr. coeff.
   .506647309000                                    -- contr. coeff.
   .200082121000                                    -- contr. coeff.
    1    1                                          -- 1d -> 1d
   .4210000000                                      -- d-exponent
  1.000000000000                                    -- contr. coeff.
*                                                   -- comment line
* Core AIMP: SQR-2P                                 -- comment line
*                                                   -- comment line
* Local Potential Parameters : (ECP convention)    -- comment line
*                            A(AIMP)=-Zeff*A(ECP)   -- comment line
M1                                                  -- M1 operator
    9                                               -- number of M1 terms
   237485.0100                                      -- M1 exponent
   24909.63500                                      -- M1 exponent
   4519.833100                                      -- M1 exponent
   1082.854700                                      -- M1 exponent
   310.5610000                                      -- M1 exponent
   96.91851000                                      -- M1 exponent
   26.63059000                                      -- M1 exponent
   9.762505000                                      -- M1 exponent
   4.014487500                                      -- M1 exponent
                                                    -- blank line
   .019335998333                                    -- M1 coeff.
   .031229360000                                    -- M1 coeff.
   .061638463333                                    -- M1 coeff.
   .114969451667                                    -- M1 coeff.
   .190198283333                                    -- M1 coeff.
   .211928633333                                    -- M1 coeff.
   .336340950000                                    -- M1 coeff.
   .538432350000                                    -- M1 coeff.
   .162593178333                                    -- M1 coeff.
M2                                                  -- M2 operator
    0                                               -- number of M2 terms
COREREP                                             -- CoreRep operator
   1.0                                              -- CoreRep constant
PROJOP                                              -- Projection operator
    1                                               -- highest ang. mom.
    8    2                                          -- 8s -> 2s
  184.666320      18.1126960                        -- 1s,2s proj. op. constants
   3459.000000                                      -- s-exponent
   620.3000000                                      -- s-exponent
   171.4000000                                      -- s-exponent
   58.53000000                                      -- s-exponent
   22.44000000                                      -- s-exponent
   6.553000000                                      -- s-exponent
   2.777000000                                      -- s-exponent
   1.155000000                                      -- s-exponent
   .018538249000   .005054826900                    -- contr. coeffs.
   .094569248000   .028197248000                    -- contr. coeffs.
   .283859290000   .088959130000                    -- contr. coeffs.
   .454711270000   .199724180000                    -- contr. coeffs.
   .279041370000   .158375340000                    -- contr. coeffs.
```

```
   .025985763000  -.381198090000                   -- contr. coeffs.
  -.005481472900  -.621887210000                   -- contr. coeffs.
   .001288714400  -.151789890000                   -- contr. coeffs.
    7    1                                         -- 7p -> 1p
  13.3703160                                       -- 2p proj. op. constant
   274.0000000                                     -- p-exponent
   70.57000000                                     -- p-exponent
   24.74000000                                     -- p-exponent
   9.995000000                                     -- p-exponent
   4.330000000                                     -- p-exponent
   1.946000000                                     -- p-exponent
   .8179000000                                     -- p-exponent
   .008300916100                                   -- cont. coeff.
   .048924254000                                   -- cont. coeff.
   .162411660000                                   -- cont. coeff.
   .327163550000                                   -- cont. coeff.
   .398615170000                                   -- cont. coeff.
   .232548200000                                   -- cont. coeff.
   .034091088000                                   -- cont. coeff.
*                                                  -- comment line
Spectral Representation Operator                   -- SR operator
Valence primitive basis                            -- SR basis specification
Exchange                                           -- Exchange operator
1stOrder Relativistic Correction                   -- mass-vel + Darwin oper.
SQR-2P                                             -- label in QRPLIB
End of Spectral Representation Operator            -- end of SR operator
```

Below is an example of an entry in the *ECP* library for a pseudo potential.

```
/Hg.ECP.Dolg.4s4p2d.2s2p1d.2e-MWB                  -- label (note the type ECP)
W. Kuechle, M. Dolg, H. Stoll, H. Preuss, Mol. Phys.-- ref. line 1
74, 1245 (1991); J. Chem. Phys. 94, 3011 (1991).   -- ref. line 2
    2.00000    2                                   -- eff. charge & highest ang.mom.
*s functions                                       -- comment line
  4  2                                             -- 4s -> 2s
  0.13548420E+01                                   -- s-exponent
  0.82889200E+00                                   -- s-exponent
  0.13393200E+00                                   -- s-exponent
  0.51017000E-01                                   -- s-exponent
  0.23649400E+00  0.00000000E+00                   -- contr. coeff.
 -0.59962800E+00  0.00000000E+00                   -- contr. coeff.
  0.84630500E+00  0.00000000E+00                   -- contr. coeff.
  0.00000000E+00  0.10000000E+01                   -- contr. coeff.
*p functions                                       -- comment line
  4  2                                             -- 4p -> 2p
  0.10001460E+01                                   -- p-exponent
  0.86645300E+00                                   -- p-exponent
  0.11820600E+00                                   -- p-exponent
  0.35155000E-01                                   -- p-exponent
  0.14495400E+00  0.00000000E+00                   -- contr. coeff.
 -0.20497100E+00  0.00000000E+00                   -- contr. coeff.
  0.49030100E+00  0.00000000E+00                   -- contr. coeff.
  0.00000000E+00  0.10000000E+01                   -- contr. coeff.
*d functions                                       -- comment line
```

```
  1  1                                              -- 1d -> 1d
  0.19000000E+00                                    -- d-exponent
  0.10000000E+01                                    -- contr. coeff.
*                                                   -- comment line
PP,Hg,78,5;                                         -- PP operator, label, # of core elec., L
1; ! H POTENTIAL                                    -- # number of exponents in the H potential
2, 1.00000000,.000000000;                           -- power, exponent and coeff.
3; ! S-H POTENTIAL                                  -- # number of exponents in the S-H potential
2,0.227210000,-.69617800;                           -- power, exponent and coeff.
2, 1.65753000,27.7581050;                           -- power, exponent and coeff.
2, 10.0002480,48.7804750;                           -- power, exponent and coeff.
2; ! P-H POTENTIAL                                  -- # number of exponents in the P-H potential
2,0.398377000,-2.7358110;                           -- power, exponent and coeff.
2,0.647307000,8.57563700;                           -- power, exponent and coeff.
2; ! D-H POTENTIAL                                  -- # number of exponents in the D-H potential
2,0.217999000,-.01311800;                           -- power, exponent and coeff.
2,0.386058000,2.79286200;                           -- power, exponent and coeff.
1; ! F-H POTENTIAL                                  -- # number of exponents in the F-H potential
2,0.500000000,-2.6351640;                           -- power, exponent and coeff.
1; ! G-H POTENTIAL                                  -- # number of exponents in the G-H potential
2,0.800756000,-13.393716;                           -- power, exponent and coeff.
*                                                   -- comment line
Spectral Representation Operator                    -- SR operator
End of Spectral Representation Operator             -- end of SR operator
```

Section 9

GUI

9.1 MOLCAS grid and geometry viewer

MOLCAS Grid/Geometry Viewer/Editor (`GV`) is an OpenGL based code for visualization of molecular orbitals and densities, density differences and spin-densities, and visualization of molecular structures (with the possibility to manipulate them on the screen), and visualization of some properties.

For installation `GV` requires headers and libraries for GLUT development. For Linux users we recommend to install libraries from *http : //www.mesa3d.org/*. Optionally it is possible to compile gv with panel support. In this case GLUI library should be installed. It is not nessesary to build `GV`. Molcas distribution provides prebuild executables for `GV`, located in ***`xbin`*** directory. If your platform is different from LINUX, you have to edit ***`xbin.cfg`*** and specify a suitable executable.

`GV` can operate with different kinds of files:

- grid files (usually with extension grid) (generated by the program `GRID_IT`),
- XYZ files,
- molden files, generated by molcas.

If a file name is specified in a command line, `GV` will determine the type of the file, and switch to the appropriate mode. If no file is specified, `GV` will search the current directory for files with a known type.

`GV` can run as a ordinary molcas module (via command 'molcas gv filename'), or as a stand alone executable.

Command line parameters for `GV`

Command	*Purpose*
filename	: name of a grid file, or XYZ file, or molden file. If this parameter is omitted program will try to find a first relevant file in the current directory.

-h	display help information
-n filename	create an empty XYZ file
-d real	set an initial size for the molecule
-f filename	set the basename of output files (.xyz, .tga, .eps, .pov). By default the basename is taken from the filename.
-s integer	Set initial size for the screen.
-g	Run gv with panels (gv should be compiled with GLUI library)
-G	similar to -g, but display panels as a separate window.

The rest of command line parameters have meaning only for displaying grid files:

Command	*Purpose*
-l real	: set initial isolevel value
-i real	: set increment value for changing isovalue
-t real	: set transparency level (between 0.0 and 1.0) for isosurfaces
-b color	: set initial background color ($white \vert gray \vert black$)
-o integer	: set the number of initial orbitals (by default: display density)
filename2	: name of a second grid file to display a density difference
-a real	: a weight of second grid. The default is -1 (to calculate density difference).
–out filename	: specify output file name for a grid constructed with -a option.

`GV` also can display a molecular structure from a native seward input. In this case a flag '-c' should be placed before the filename.

For an advance configuration of `GV` one can save the current configuration (see below), and edit the ASCII formatted configuration files. The configuration files are located in the subdirectory *`.molcasgv`* in user home directory. File *`gv.rc`* contains global settings, like colors, transparency, etc. File *`atoms.rc`* contains sizes and colors for each element.

The main control of `GV` can be done by the mouse: the left mouse button is used to rotate the grid or the molecule, right mouse button is used for the on-screen menu. The major operations from menus can also be performed by hot-keys. Note, that behavior of `GV`, e.g. on-screen menus and hot-keys is different for the different types of visualized files.

Some keys like PageUp/PageDown, plus/minus have different meaning in different modes, with a general rule - plus/minus changes a value, PageUp/PageDown changes a property from a list.

Note for Mac users. To press functinal keys (F1-F10) you have to use 'fn' button. Insert button (missing on Mac keyboard) can be emulated by 'I' key.

General HotKeys.

Command	*Purpose*
Ctrl-Q	Quit
F10	Exit (exit implies possible back up of edited files).
Escape	unset different editing modes
F1	Display the help screen (with a list of hot-keys)
q/Q	decrease/increase quality of drawing
F2	save INPORB file or XYZ file
F9	Save current settings
Shift+F9	Edit colors for background, labels and orbitals.
r/R/g/G/b/B	Interactively change RGB code for the background, labels, and orbitals (selected by Shift-F9 key)
F5	Print screen in tga format. The output files will be named as `filename000.tga`
Shift+F5	Print screen in PostScript format. The output files will be named as `filename000.eps`
P	Print screen in PostScript (level 2) format.
p	Save povray file.
z/Z	Zoom the grid or the molecule
x	maximize the screen
Home	Move the molecule to the center of the screen
Up/Down/Left/Right	Move the position of the picture on screen (with shift - make bigger steps)
l	Switch to/from the mode: move the light position with the mouse.
w	Switch between greyscale and colored picture
m	Start/stop animation
W	Switch to Tee time mode

Hot keys in Grid Mode.

Command	*Purpose*
+/-	increase/decrease the isosurface value (the step and initial value can be modified by command line parameters)
PageUp/PageDown	display next/previous orbital. In multiview mode (F3 has been pressed), use magnify glass.

t/T	change transparancy level
f/i/1/2(a)/3/s/d	change the type of the current orbital to frozen, inactive, RAS1, RAS2, RAS3, secondary, deleted.
Space/middle mouse	change the type of the current orbital (by loop)
F2	save INPORB file (file will be saved as *`filename.GvOrb`*)
F3	Switch to/from multiview mode. In multiview mode the orbital type is shown by different backgrounds (rainbow colors). User can modify the type of orbitals, by pressing middle mouse button (or Space).
F4	Enter an isovalue, or an orbital number, or create a filter (for more information check tuturial for `GV`).
Delete	Hide the orbital from the list
Insert	Restore all hidden orbitals

If `GV` is used to display a molecule, it is possible to make a selection of an atom, a bond, an angle, or a dihedral angle by clicking (by left mouse button) on the atom. First selected atom will be displayed with a blue colored net, the rest (for bond and angle) with a magenta colored net. Using hot keys it is possible to make modification of the selected part of the structure. In addition to selection (note that only 4 atoms can be selected), it is possible to mark a group of atoms. Marking is made either by mouse (is Shift button is hold), or by keyboard (F7 button). Marked atoms can be modified as a single unit - e.g. rotated, moved. To remove current selection - press middle mouse button or space. If user has both selected and marked atoms - the first press of space button will unselect 'selected', and the second one - unselect 'marked' atoms.

Hot keys in Coordinate mode.

Command	*Purpose*
F2	Save coordinates (file will be saved as *`filename000.xyz`*)
Shift-F2	Save coordinates (overwrite the file)
left mouse	select an atom by clicking on it.
left mouse + shift	mark an atom
F3	display menu with fragments.
M	if Mopac is installed - optimize geometry (you might need to modify *`sbin/runmopac`* script and fix the location of Mopac installation)

Hot keys in if no atoms are selected

Command	*Purpose*
Insert	add an atom, or last inserted fragment

End	add dummy atoms (reference points) on the direction of axis.
F7	Mark hydrogen atoms in the molecule
F8	analyze the symmetry of the molecule and display symmetry elements.
Delete	delete dummy atoms
Home	Move the molecule to the center of screen

Hot keys in Selection mode (1 atom is selected)

Command	*Purpose*
Space/middle mouse	remove selection.
F3	display a menu with fragments, to be inserted close to selected atom
Insert	add an atom (or last selected fragment) near selected atom
Delete	delete selected atom
Home	place the origin to the position of selected atom
PageUp/PageDown	Change selected atom to one from the list (H,C,N,O,F,S,Cl)
F4 or =	envoke an edit box, where you can type an element name for selected atom.
F7	Mark atoms connected to the selected atom.
F8	apply Inversion symmetry around selected atom

Hot keys in Selection mode for bond (2 atoms are selected)

Command	*Purpose*
Space/middle mouse	remove selection.
Insert	create a bond between selected atoms
Delete	delete the bond between selected atoms
PageUp/PageDown	change the type of the bond between atoms
+/-	change the distance between atoms. Note that first selected atom (blue) will move.
F4 or =	envoke an edit box, where you can type an interatomic distance
F7	Mark all connected atoms around the first atom in a selected bond into a group.
F8	apply C_2 symmetry around an axis specified by selected atoms.

F6	Watch the value of selected bond

Hot keys in Selection mode for angle (3 atoms are selected)

Command	*Purpose*
Space/middle mouse	remove selection.
+/-	change the angle between selected atoms. Note that first selected atom (blue) will move.
PageUp/PageDown	change the angle according to 'standard' angle values (by loop)
F4 or =	envoke an edit box, where you can type an angle value (or, you may type the value directly)
F8	apply mirror symmetry around an plain specified by selected atoms.
F6	Watch the value of selected angle

Hot keys in Selection mode for dihedral angle (4 atoms are selected)

Command	*Purpose*
Space/middle mouse	remove selection.
+/-	change the dihedral angle between selected atoms. Note that first selected atom (blue) will move.
F4 or =	envoke an edit box, where you can type a dihedral angle value (or, you may type the value directly)
F6	Watch the value of selected angle

Hot keys in Molden mode for a orbital file (e.g. scf.molden)

Command	*Purpose*
PageUp/PageDown	Display charges

Hot keys in Molden mode for a frequency file

Command	*Purpose*
PageUp/PageDown	load next/previous vibration mode
F3	draw graphical information in a separate window.
+/-	change the speed of vibrations

Hot keys in Molden mode for a geometry file

Command	*Purpose*
PageUp/PageDown	load next/previous geometry
F3	draw graphical information in a separate window.
Home	show initial structure
End	show the resulting structure

9.2 Writing GV/MOLDEN input

By default the `SCF`, `RASSCF`, `CASVB`, `SLAPAF`, and `MCLR` modules generate input in `GV` and Molden format. The `SCF` and `RASSCF` modules generate input for molecular orbital analysis, `CASVB` for valence bond orbital analysis, `SLAPAF` for geometry optimization analysis, minimum energy paths, Saddle optimization paths and IRC TS analysis, and the `MCLR` module generates input for analysis of harmonic frequencies. For further details with respect to Molden consult

http://www.caos.kun.nl/ shaft/molden/molden.html.

The generic name of the input file is MOLDEN. However, the actual name is different for the nodes as a reflection on the data generated by each module. Hence, the actual names for MOLDEN in each module are

- SCF module: $Project.scf.molden
- GUESSORB module: $Project.guessorb.molden
- RASSCF module: $Project.rasscf.molden for the state-averaged natural orbitals, and $Project.rasscf.x.molden for the state-specific natural spin orbitals, where x is the index of a CI root.
- CASVB module: $Project.casvb.molden
- SLAPAF module: $Project.geo.molden for geometry optimizations, $Project.mep.molden for minimum energy paths, $Project.saddle.molden for Saddle method TS optimizations and $Project.irc.molden for IRC analysis of a TS.
- MCLR module: $Project.freq.molden

Part III

Installation Guide

Section 10

Installation

The present installation guide describes the necessary steps for installing and tailoring MOLCAS. It also describes the steps for applying updates whenever necessary.

The installation procedure can be reduced to a few simple steps:

1. Extract the contents of the tar
2. Configure the package
3. Build the package
4. Build GUI and documentation (optional)
5. Make the package generally available.

10.1 Prerequisites

10.1.1 Prerequisite hardware

MOLCAS can be built on any hardware that runs under a UNIX operating system. Some of these variants of hardware and software have been tested by us, and you should not have any problems to install MOLCAS on any of these. For other platforms you must, most likely, put some effort into this. In many cases the only effort on your part is setting some compiler flags, paths to system software etc. For a list of the platforms where we have successfully installed MOLCAS see our homepage: *`http://www.molcas.org`*.

To load the executables resident, sufficient memory is required. In addition, the programs are enabled to allocate work space dynamically. To avoid excessive paging we recommend that your machine should be equipped with at least 2 GB of memory per running application. Note, that MOLCAS will run faster with more memory.

To install MOLCAS you should have about 300 MB of free disk space. To run the verification tests of MOLCAS you should have a scratch disk with up to 1 GB of free disk space, depending on the suite you run. For the “small” set about 400 MB will suffice. To perform larger calculations, ample amount of scratch disk space is necessary. The exact amount varies with the type of systems studied, but a general recommendation is at least 4 GB of disk space, per production run.

10.1.2 Prerequisite software

MOLCAS is never prebuilt by us, you have to build it to install. Thus, make certain that the necessary system software is available. To build molcas you need:

- A Fortran 77 compiler
- A Fortran 90 compiler (for some modules)
- A C compiler
- The GNU make facility. See URL *`http://www.gnu.org`* and navigate to the gnumake page or go directly to *`http://www.gnu.org/gnulist/production/make.html`*.
- Perl (5.008 or higher)
- uuencode (on some Linux installations it is a part of sharutils)

To install *MOLCAS* under MS Windows (98/NT/XP) one should install Cygwin (freeware from RedHat Inc., which can be downloaded from www.cygwin.org). It is highly recommended to install almost all components of Cygwin, especially development and shell utilities. Cygwin is available for different versions of MS Windows and includes the GNU gcc/g77 compiler. After installation you will get an 'UNIX-like' command prompt, where you can install *MOLCAS* in the same way as under Linux.

Installation of *MOLCAS* under MacOS requires installation of Darwin and a Fortran compiler. These programs could be downloaded from http://developer.apple.com. For PPC architecture it is possible to use commercial xlf/xlc compilers.

Graphical User Interface codes in molcas requires an additional system software, including OpenGL and glut library. However, in most of the cases there is no need to install these libraries, since executables for GUI are included into the distribution, or they can be downloaded from Molcas webpage.

10.1.3 Preparing the installation

In order to install *MOLCAS* you need to choose a directory where *MOLCAS* driver is to be installed. Driver is a small script (created during the installation), which executes scripts and programs form the *MOLCAS* package and it must be located in any directory included into the `PATH`.

Driver script *`molcas`* uses the value of the environment variable `MOLCAS` to identify which version to use. The major advantage with this mechanism is that it is easy to switch between different versions of *MOLCAS* by simply changing the environment variable `MOLCAS`. However if the current directory is a subdirectory (up to 3-rd level) of *MOLCAS* tree, this version will be used.

MOLCAS itself can be located in any place on the disk. The installation can be done by root, or by unprivileged user. In the later case you can copy molcas driver script to an appropriate location, e.g. *`/usr/local/bin`* after the installation.

All files are contained in a tar archive file with the name *`molcas74.tar.gz`*, you need to uncompress the file with the command gunzip molcas74.tar.gz, and untar the package with tar -xvf molcas74.tar.

10.2 Configuring *MOLCAS*

Before you can build *MOLCAS* you have to configure it. Most common platforms have been setup by the *MOLCAS* team, so for serial installation with default settings for compiler and compiler flags configuration of molcas can be done by run the `configure` script:

```
./configure
```

If *MOLCAS* is built in the first time, `configure` will suggest to run setup script, to analyse the system and to give some suggestions about the available compilers, flags, which are specific for parallel installation.

Setup can be run manually by a command:

```
./configure -setup
```

When configuration is finished, you should review the log file `configure.log` to see if everything is ok. There is no harm in running the configuration script even if it should fail, you simply rerun it with correct parameters.

If configuration step was not successful, you probably is missing some prerequisite software, or this software is located in unusual location on the disk. In the later case you might need to update your PATH, or use flag `-path` in `configure`.

The configuration step is described in details in the Molcas Programming guide.

10.3 Building *MOLCAS*

When the configuration step (section 10.2) is completed successfully, you can build *MOLCAS*. This is simply done by typing `make` in the *MOLCAS* root directory. It is recommended that you save the output from `make` in a log file for tracing of potential problems.

```
make > make.log 2>&1
```

When *MOLCAS* is being compiled some compilers give a lot of warnings. These are not serious in most cases. We are working on eliminating them, but the job is not yet completely finished.

At the final step of make, a simple test verifying a general functionality of *MOLCAS* is executed.

10.3.1 Building GUI and documentation

Normally, there is no need to build GUI used in *MOLCAS* since we provide executables for most common platforms. These executables are located in `xbin` directory. And user should only edit `xbin.cfg` file to select proper executable for his platform. Alternatively, these executables can be downloaded from *MOLCAS* homepage.

When the building step (section 10.3) is completed successfully, you can run script config.extra to check system related commands and libraries which are needed to build GUI and documentation. If the script reports that some software should be installed, you can install it, and rerun config.extra. Alternatively, you can download executables for GUI and the manual from molcas webpage.

If config.extra reported that all prerequisite software is installed, you can type make extra to build GUI and the manual.

Also, you can use the following make targets:

- make gv : to build gv (geometry and grid viewer) without panel support
- make gv_glui: to build gv with panel support
- make ming: to build MING (molcas input generator) and MolGUI (molecular builder) (note that source code for MING and MolGUI might be not included into distribution).
- make doc: to build the manual in various formats: PostScript, PDF, HTML
- make doc_ps: to build only PostScript version of the manual

10.3.2 Verifying the *MOLCAS* installation

After a successful build of *MOLCAS* you should verify that the various modules run correctly. Directory *Test/* contains test inputs for *MOLCAS*. Use command molcas verify [parameters] to start verification. Running this command without parameters you will check main modules and features of *MOLCAS* and this option we recommend for verifying the installation. You can also specify a keyword as argument that translates into a sequence of test jobs, or you can specify a list of test jobs yourself. Below are a few examples:

molcas verify — will run the standard set of tests.

molcas verify performance — will run performance tests. This will take a long time and is normally not recommended.

To generate a report after performance tests you should execute a command molcas timing. The report is now located in the file *Test/timing/user.timing*. The results of benchmark tests for some machines are collected at the location
http://www.molcas.org/benchmark.html At the completion of the test suite a log of the results is generated in the file *Test/Results*. If installation was performed by another user (e.g. root), you can redefine the location of output files by adding the flag -path PATH. Each test job is signaled as either ok of failed. If there are any failed jobs, the outputs are saved in *Test/Failed_Tests*. Each test job tests for a resulting checksum for the modules tested. This checksum is typically the energy for a wavefunction program such as RASSCF, whereas other types of codes use other checksums.

The checksums will not match exactly with our reference values since different machines use different arithmetics. We have tried to make the acceptable tolerances as small as possible and at the same time make all tests pass successfully. It might be the case that your particular platform will produce one or more results that are just outside our tolerances, and in such a case the test is most likely ok.

More information about usage of verify can be found in the Programming Guide.

10.4 Installing and running in parallel environments

Installation of *MOLCAS* for execution in multi-processor environments can be a bit more involved than the standard installation, this chapter considers those particulars not covered previously.

Currently the parallel version of *MOLCAS* has been tested under Linux clusters (with MPICH) and under Fujitsu Solaris.

Use ./configure -setup command to see the suggestions about recommended flags for parallel installation.

10.4.1 Overview of the procedure

In the simplest case, the parallel version of *MOLCAS* may be installed simply by specifying an appropriate message-passing system as an argument to *`configure`*. For example:

```
cd $MOLCAS
./configure -parallel mpich
make
```

Parallel execution of *MOLCAS* is then achieved by exporting the system variable CPUS:

```
CPUS=4; export CPUS
```

and continuing as usual.

More likely, some individual tailoring will be required, the following summarizes the necessary steps:

1. Choose message passing model (likely candidates are: mpi,mpich,ch_p4,lapi,lapi+mpi).
2. Check location of necessary libraries and commands, as specified in *`$MOLCAS/Symbols`*.
3. Install (and test) the Global Arrays package (see below).
4. Check the command for executing binaries in parallel, as specified by $RUNBINARY in *`$MOLCAS/molcas.rte`*.
5. Install (and test) *MOLCAS*.

Provided that steps 1–4 can be successfully accomplished, the installation of *MOLCAS* itself is unlikely to present many difficulties.

The remainder of this chapter is devoted to a more detailed description of *MOLCAS*'s parallel setup.

10.5 Utilities and special considerations

10.5.1 Global Arrays

The parallelization of *MOLCAS* relies on the "Global Arrays" toolkit, developed by Jarek Nieplocha and coworkers at the Pacific Northwest National Laboratory:

http://www.emsl.pnl.gov:2080/docs/global/ga.html

A version of this software has been included with the *MOLCAS* distribution. We feel that this simplifies the installation procedure for the majority of cases, as well as eliminating compatibility problems between *MOLCAS* and different versions of GA. However, in many cases it can be advantageous to use a newer version, especially if a recent platform is being used.

The installation instructions may be found at the Global Arrays home page and in the file *g/README*. Verification tests may be found in *g/global/testing/*. Note that any problems with installation or other issues specific to GA are best resolved by contacting the GA authors directly, rather than the *MOLCAS* group.

10.5.2 "MPICH" – public-domain implementation of MPI

http://www-unix.mcs.anl.gov/mpi/mpich/

If you use the MPICH tools, then it comes with FORTRAN 77 and C wrappers for the GNU compilers (mpif77 and mpicc respectively). Although it seems easier just to use these wrappers instead of specifying the locations of the libraries and include files manually, it is strongly recommended to do it manually anyway.

In other words, a MPICH installation should be treated as it were a normal MPI installation.

10.5.3 Distributed-memory architectures with MPI

For a cluster of workstations (e.g. a Beowulf cluster), take this issues into consideration:

- Workstations must be binary compatible.
- Use TCGMSG ("parallel" script in GA) or MPICH.
- Process-group file for "parallel" script with different workstation hostnames.

A few comments on the last item. MPICH needs a file with a list of the nodes the job at hand is allowed to use. At default the file is static and located in the MPICH installation tree. This will not work on a workstation cluster, though, because then all jobs would use the same nodes.

Instead the queue system sets up a temporary file, which contains a list of the nodes to be used for the current task. You have to make sure that this filename is transfered to $mpirun. This is done with the '-machinefile' flag. On a Beowulf cluster using PBS as queue system the $RUNBINARY variable in *$MOLCAS/molcas.rte* should look something like:

RUNBINARY='/usr/local/mpich/bin/mpirun -machinefile $PBS_NODEFILE -np $CPUS $program'

Section 11

Maintaining the package

11.1 Tailoring

MOLCAS, as shipped, is configured with some default settings. You can change some of these easily. You can change default settings used in *MOLCAS* (like memory usage, default scratch area, policy in saving files, etc.) by editing molcas resource file: global resource file *$MOLCAS/molcasrc* or user resource file *$HOME/molcasrc*.

11.1.1 Dynamic memory

Most modules in *MOLCAS* utilize dynamic memory allocation. The amount of memory each module allocate is controlled by the environment variable `MOLCASMEM`. The amount of memory allocated is

- `MOLCASMEM` is undefined — 256MB of memory is allocated.
- `MOLCASMEM=nn` — `nn`MB is allocated. If this amount cannot be allocated, the module stops.

11.1.2 Disk usage

Today many workstations utilize 64-bit integers and addressing. However, old UNIX workstations and PC's had 32-bit integers resulting in a file size limit of 2GB. To circumvent these limitations, the I/O routines of *MOLCAS* support multifile files, where a "file" is in reality a logical file consisting of several physical files. The size limit of these physical files is controlled by the environment variable `MOLCASDISK` according to

- `MOLCASDISK` is undefined — The modules will use a 2GB size of the physical files. This might be the appropriate setting for machines with 32-bit addressing.
- `MOLCASDISK=nn` — The modules will use a `nn`MB size of the physical files.

To use files with a size bigger than 2GB *MOLCAS* should be compiled as 64-bit executable.

11.1.3 Improving performance

MOLCAS is shipped with a number of default setup files located in directory *cfg/*. The defaults in these files are set to a fairly safe level, but not necessary optimal. What you can change to improve performance is

- Compiler flags
- Mathematical (blas) libraries

The simplest way to set up optimization level, and/or compile *MOLCAS* with various BLAS libraries is to use configure -setup. This interactive script helps to make a proper selection of flags for improvement of *MOLCAS* performance.

If you do decide to try to improve the performance we recommend that you create a new setup file, for example, *cfg/local.cfg* and modify this file. It is not unlikely that your attempts to optimize the codes will lead you to a case where some modules work and others do not. In such a scenario it can be fruitful to have two copies of *MOLCAS*, one "safe" where all modules work and one "fast" where some modules do not function properly.

Changing the compiler flags is the easiest. Using the most aggressive optimization flags do sometimes lead to problems for some of the modules. We have tried to choose an optimization level that yields functioning code, but still reasonable fast. For some systems there is a predefined set of compiler flags for aggressive optimization. To compile *MOLCAS* with these flags you should run configure with flag -speed fast.

For some platforms you can utilize the vendor blas libraries. This will certainly yield better performance, but may not work on all platforms.

During configuration of *MOLCAS* it is possible to specify an external BLAS/LAPACK library. Use a flag -blas TYPE to specify the type of BLAS libary: lapack (for a standard lapack library), Goto (for GotoBLAS), Atlas (for ATLAS), MKL (for Intel MKL). In case of Goto, Atlas and MKL, you should also specify a flag -blas_dir DIR pointing to the library location. For example, to configure *MOLCAS* with Intel MKL library, you should issue a command ./configure -compiler intel -blas MKL -blas_dir /opt/intel/mkl/32/lib

It is also possible to make a manual installation of a vendor supplied BLAS library. One should issue commands molcas uninstall blas_util, molcas uninstall essl_util and molcas uninstall lapack_util to remove BLAS/LAPACK related directories from the molcas source code, then export XLIB variable to set the location of blas library, e.g. XLIB="-lblas"; export XLIB, and finally reconfigure and build molcas. If the library is in a nonstandard location you may have to issue the command XLIB="-Lpath_to_lib -lblas"; export XLIB. Alternatively, define XLIB in the system specific configuration file.

After making changes to the setup files you have to issue the commands make veryclean, ./configure and make in the *MOLCAS* root directory. It is highly recommended to run the verification suite after any changes in configuration file.

11.2 Applying patches

All program systems do contain bugs and *MOLCAS* is certainly no exception. We prepare patches for all problems as soon as we identify and fix the problem. You can get these patches from our web server in an easy and automatic way.

For important updates we provide Service Packs. A service pack is a shell script, which makes a backup of your current *MOLCAS* installation, and installs updates.

it is also possible to download and install individual patches by issuing the command molcas getpatch. The patchlevel of your installation is identified, all newer patches are downloaded and applied. Note that this feature requires Perl 5.008 or higher.

If your computer is not connected directly to Internet (or getpatch failed), you still can update your version of *MOLCAS* in a rather easy way. Command molcas getpatch -file produces a file ***getpatch.html*** which contains all information about patch level in your *MOLCAS* installation. This file could be transfered to another computer. Open the file with your favourite Web-browser, and follow instructions to download file ***update.tar***. Then place ***update.tar*** to *MOLCAS* root directory, untar it, and execute the command ./apply.

It is also possible to check availability of new patches without installing them, by using a flag -query: e.g. command molcas getpatch -u -q reports patches, which are missing for current installation in respect to 'unsafe' lavel of patches.

After applying the patches you need to rebuild the package by issuing the command make in the *MOLCAS* root directory.

When you have applied a patch you will see a new directory appear, ***patch/7.4.x/***. To revert a patch you can issue a command molcas revert patch XXX, where XXX is a number of the patch. Once again it is necessary to redo make.

11.2.1 Local modifications

MOLCAS is shipped with source code so you can make modifications yourself. You are, of course, responsible for the correctness of any such modification.

If you do make changes/additions to the source code that you feel is of interest to other users, we encourage you to make these available. Perhaps the best mechanism is to use the bulletin board on out homepage: ***http://www.molcas.org***.

Check Molcas Programming Guide for a detailed description of development and distribution of modified code in *MOLCAS* .

Bibliography

[1] B. O. Roos, P.-Å. Malmqvist, and L. Gagliardi. Heavy element quantum chemistry - the multiconfigurational approach. In (E. Brändas and E. Kryachko), editors, *Fundamental World of Quantum Chemistry*, page 425. Kluwer Acad. Publ., Dordrecht, Netherlands, 2003.

[2] B. O. Roos, V. Veryazov, and P.-O. Widmark. Relativistic ano type basis sets for the alkaline and alkaline earth atoms applied to the ground state potentials for the corresponding dimers. *Theor. Chim. Acta*, 111:345, 2004.

[3] B. O. Roos, R. Lindh, P.-Å. Malmqvist, V. Veryazov, and P.-O. Widmark. Main group atoms and dimers studied with a new relativistic ano basis set. *J. Phys. Chem. A*, 108:2851, 2004.

[4] B. O. Roos, R. Lindh, P.-Å. Malmqvist, V. Veryazov, and P.-O. Widmark. New relativistic ano basis sets for transition metal atoms. *J. Phys. Chem. A*, 109:6575–6579, 2005.

[5] B. O. Roos, R. Lindh, P.-Å. Malmqvist, V. Veryazov, and P.-O-Widmark. New relativistic ano basis sets for actinide atoms. *Chem. Phys. Letters*, 409:295–299, 2005.

[6] B. O. Roos, R. Lindh, P.-Å. Malmqvist, V. Veryazov, P.-O. Widmark, and A. C. Borin. New relativistic ano basis sets for lanthanide atoms with applications to the ce diatom and luf$_3$. *J. Phys. Chem. A*, in press, 2008.

[7] F. Aquilante and R. Lindh and T. B. Pedersen. Cholesky decomposition as a tool for generating unbiased auxiliary basis sets for accurate two-electron integral approximations. *Submitted.*

[8] F. Aquilante, P.-Å. Malmqvist, T. B. Pedersen, A. Ghosh, and B. O. Roos. Cholesky decomposition based multiconfigurational second order perturbation theory (CD-CASPT2): Application to the spin state energetics of Co$^{\mathrm{III}}$(diiminato)(NPh). 4:694–702, 2008.

[9] F. Aquilante, T. B. Pedersen, B. O. Roos, A. Sánchez de Merás, and H. Koch. Accurate ab inition density fitting for multiconfigurational self-consistent field methods. *J. Chem. Phys.*, 129:024113, 2008.

[10] G. Karlström, R. Lindh, P.-Å. Malmqvist, B. O. Roos, U. Ryde, V. Veryazov, P.-O. Widmark, M. Cossi, B. Schimmelpfennig, P. Neogrady, and L. Seijo. Molcas: a program package for computational chemistry. *Computational Material Science*, 28:222, 2003.

[11] V. Veryazov, P. -O. Widmark, L. Serrano-Andrés, R. Lindh, and B. O. Roos. *Int. J. Quantum Chem.*, 100:626, 2004.

[12] B. O. Roos. Accurate calculations and calibration. In B. O. Roos, editor, *Lecture Notes in Quantum Chemistry, European Summer School in Quantum Chemistry, in: Lecture Notes in Chemistry, 58*, page 177. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[13] J. Finley, P.-Å. Malmqvist, B. O. Roos, and L. Serrano-Andrés. The multi-state CASPT2 method. *Chem. Phys. Letters*, 288:299–306, 1998.

[14] J. D. Watts, J. Gauss, and R. J. Bartlett. *J. Chem. Phys.*, 98:8718, 1983.

[15] P. Neogrády and M. Urban. *Int. J. Quantum Chem.*, 55:187, 1995.

[16] U. Wahlgren. The effective core potential method. In B. O. Roos, editor, *Lecture Notes in Quantum Chemistry, European Summer School in Quantum Chemistry, in: Lecture Notes in Chemistry, 58*, page 413. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[17] L. Seijo and Z. Barandiarán. The ab initio model potential method: A common strategy for effective core potential and embedded cluster calculations. In J. Leszczynski, editor, *Computational Chemistry: Reviews of Current Trends*, volume 4, page 55. World Scientific, Singapur, 1999.

[18] H. Partridge, S. R. Langhoff, and C. W. Bauschlicher. Electronic spectroscopy of diatomic molecules. In S. R. Langhoff, editor, *Quantum Mechanical Electronic Structure Calculations with Chemical Accuracy*, page 209. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[19] P. R. Taylor. Molecular symmetry and quantum chemistry. In B. O. Roos, editor, *Lecture Notes in Quantum Chemistry, European Summer School in Quantum Chemistry, in: Lecture Notes in Chemistry, 58*, page 89. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[20] M. R. A. Blomberg, P. E. M. Siegbahn, and B. O. Roos. *Mol. Phys.*, 47:127, 1982.

[21] B. O. Roos. The complete active space self-consistent field method and its applications in electronic structure calculations. In K. P. Lawley, editor, *Advances in Chemical Physics; Ab Initio Methods in Quantum Chemistry - II*, chapter 69, page 399. John Wiley & Sons Ltd., Chichester, England, 1987.

[22] R. Pou-Amérigo, M. Merchán, I. Nebot-Gil, P.-Å. Malmqvist, and B. O. Roos. The chemical bonds in CuH, Cu_2, NiH, and Ni_2 studied with multiconfigurational second order perturbation theory. *J. Chem. Phys.*, 101:4893, 1994.

[23] K. Andersson and B. O. Roos. Excitation energies in the nickel atom studied with the complete active space SCF method and second-order perturbation theory. *Chem. Phys. Letters*, 191:507, 1992.

[24] G. Herzberg. *Molecular Spectra and Molecular Structure.* Van Nostrand, New York, 1939-1966.

[25] P. R. Taylor. Accurate calculations and calibration. In B. O. Roos, editor, *Lecture Notes in Quantum Chemistry, European Summer School in Quantum Chemistry, in: Lecture Notes in Chemistry, 58*, page 325. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[26] R. González-Luque, M. Merchán, and B. O. Roos. A theoretical determination of the dissociation energy of the nitric oxide dimer. *Theor. Chim. Acta*, 88:425, 1994.

[27] M. Peric, B. Engels, and S. D. Peyerimhoff. Theoretical spectroscopy on small molecules. In S. R. Langhoff, editor, *Quantum Mechanical Electronic Structure Calculations with Chemical Accuracy*, page 261. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[28] T. Helgaker. Optimization of minima and saddle points. In B. O. Roos, editor, *Lecture Notes in Quantum Chemistry, European Summer School in Quantum Chemistry, in: Lecture Notes in Chemistry, 58*, page 295. Springer-Verlag, Berlin, Heidelberg, New York, 1992.

[29] R. Lindh, A. Bernhardsson, and M. M. Schütz. Force-constant weighted redundant coordinates in molecular geometry optimizations. *Chem. Phys. Letters*, 303:719, 1999.

[30] M. Rubio, M. Merchán, E. Ortí, and B. O. Roos. A theoretical study of the electronic spectrum of naphthalene. *Chem. Phys.*, 179:395, 1994.

[31] L. Serrano-Andrés, M. Merchán, I. Nebot-Gil, R. Lindh, and B. O. Roos. Towards an accurate molecular orbital theory for excited states: Ethene, butadiene, and hexatriene. *J. Chem. Phys.*, 98:3151, 1993.

[32] L. Serrano-Andrés and B. O. Roos. Theoretical study of the absorption and emission spectra of indole in gas and in a solvent. *J. Am. Chem. Soc.*, 118:185, 1996.

[33] C. S. Page, M. Merchán, L. Serrano-Andrés, and M. Olivucci. A theoretical study of the low-lying excited states of trans- and cis-urocanic acid. *J. Phys. Chem. A*, 103:9864, 1999.

[34] R. Pou-Amérigo, M. Merchán, and E. Ortí. *J. Chem. Phys.*, 110:9536, 1999.

[35] C. E. Blom and A. Bauder. *Chem. Phys. Letters*, 88:55, 1982.

[36] V. Molina and M. Merchán. Theoretical analysis of the electronic spectra of benzaldehyde. *J. Phys. Chem. A*, 105:3745, 2001.

[37] F. Ford, T. Yuzawa, M. S. Platz amd S. Matzinger, and M. P. Fülscher. The rearrangement of dimethylcarbene to propene: A laser flash photolysis and *ab initio* study. *J. Am. Chem. Soc.*, 120:4430, 1998.

[38] T. J. Lee and P. R. Taylor. *Int. J. Quantum Chem.*, 23:199, 1989.

[39] T. J. Lee and G. E. Scuseria. Achieving chemical accuracy with coupled-cluster theory. In S. R. Langhoff, editor, *Quantum Mechanical Electronic Structure Calculations with Chemical Accuracy*, page 47. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[40] S. Matzinger and M. P. Fülscher. Methyl substitution in carbenes. A theoretical prediction of the singlet-triplet energy separation of dimethylcarbene. *J. Phys. Chem.*, 99:10747, 1995.

[41] D. J. Tozer, R. D. Amos, N. C. Handy, B. O. Roos, and L. Serrano-Andrés. Does funcional theory contribute to the understanding of excited states of unsaturated organic compounds? *Mol. Phys.*, 97:859–868, 1999.

[42] B. O. Roos, M. P. Fülscher, Per-Åke Malmqvist, M. Merchán, and L. Serrano-Andrés. Theoretical studies of electronic spectra of organic molecules. In S. R. Langhoff, editor, *Quantum Mechanical Electronic Structure Calculations with Chemical Accuracy*, page 357. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1995.

[43] B. O. Roos, K. Andersson, M. P. Fülscher, P.-Å. Malmqvist, L. Serrano-Andrés, K. Pierloot, and M. Merchán. Multiconfigurational perturbation theory: Applications in electronic spectroscopy. In I. Prigogine and S. A. Rice, editors, *Advances in Chemical Physics: New Methods in Computational Quantum Mechanics, Vol. XCIII:219.* John Wiley & Sons, New York, 1996.

[44] L. Serrano-Andrés, M. Fülscher, B. O. Roos, and M. Merchán. Theoretical study of the electronic spectrum of imidazole. *J. Chem. Phys.*, 100:6484, 1996.

[45] L. Serrano-Andrés. *Theoretical Study of the Electronic Spectra of Organic Systems.* PhD thesis, University of Valencia, Dept. Química Física, Dr. Moliner 50, Burjassot,E-46100 Valencia, Spain, 1994.

[46] L. Serrano-Andrés, M. Merchán, M. Fülscher, and B. O. Roos. A theoretical study of the electronic spectrum of thiophene. *Chem. Phys. Letters*, 211:125, 1993.

[47] K. Kaufmann, W. Baumeister, and M. Jungen. *J. Phys. B: At. Mol. Opt. Phys.*, 22:2223, 1989.

[48] M. P. Fülscher and B. O. Roos. The excited states of pyrazine: A basis set study. *Theor. Chim. Acta*, 87:403, 1994.

[49] K. Andersson, P.-Å. Malmqvist, B. O. Roos, A. J. Sadlej, and K. Wolinski. Second-order perturbation theory with a CASSCF reference function. *J. Phys. Chem.*, 94:5483, 1990.

[50] K. Andersson, P.-Å. Malmqvist, and B. O. Roos. Second-order perturbation theory with a complete active space self-consistent field reference function. *J. Chem. Phys.*, 96:1218, 1992.

[51] K. Andersson. *Multiconfigurational Perturbation Theory.* PhD thesis, University of Lund, Theor. Chemistry, Chem. Center, P.O.B. 124,S-221 00 Lund, Sweden, 1992.

[52] B. O. Roos and K. Andersson. Multiconfigurational perturbation theory with level shift — the Cr_2 potential revisited. *Chem. Phys. Letters*, 245:215, 1995.

[53] N. Forsberg and P.-Å. Malmqvist. Multiconfiguration perturbation theory with imaginary level shift. *Chem. Phys. Letters*, 274:196, 1997.

[54] B. O. Roos, K. Andersson, M. P. Fülscher, L. Serrano-Andrés, K. Pierloot, M. Merchán, and V. Molina. Applications of level shift corrected perturbation theory in electronic spectroscopy. *J. Mol. Struct. Theochem*, 388:257, 1996.

[55] M. P. Fülscher, L. Serrano-Andrés, and B. O. Roos. A theoretical study of the electronic spectra of adenine and guanine. *J. Am. Chem. Soc.*, 119:6168, 1997.

[56] L. Serrano-Andrés and M. P. Fülscher. Theoretical study of the electronic spectroscopy of peptides. 1. The peptidic bond: Primary, secondary, and tertiary amides. *J. Am. Chem. Soc.*, 118:12190, 1996.

[57] M. Merchán, E. Ortí, and B. O. Roos. Theoretical determination of the electronic spectrum of free base of porphin. *Chem. Phys. Letters*, 226:27, 1994.

[58] L. Serrano-Andrés and B. O. Roos. A theoretical study of the indigoid dyes and their chromophore. *Chem. Eur. J.*, 3:717, 1997.

[59] K. Pierloot, E. Van Praet, L. G. Vanquickenborne, and B. O. Roos. Systematic ab initio study of the ligand field spectra of hexacyanometalate complexes. *J. Phys. Chem.*, 97:12220, 1993.

[60] K. Pierloot, J. O. A. De Kerpel, U. Ryde, and B. O. Roos. A theoretical study of the electronic spectrum of plastocyanin. *J. Am. Chem. Soc.*, 119:218, 1997.

[61] K. Pierloot, E. Tsokos, and B. O. Roos. 3p-3d intershell correlation effects in transition metal ions. *Chem. Phys. Letters*, 214:583, 1993.

[62] K. Andersson. Different forms of the zeroth-order Hamiltonian in second-order theory with a complete active space self-consistent field reference function. *Theor. Chim. Acta*, 91:31, 1995.

[63] M. Merchán and R. González-Luque. *Ab Initio* study on the low-lying excited states of retinal. *J. Chem. Phys.*, 106:1112, 1997.

[64] L. Serrano-Andrés, M. Merchán, B. O. Roos, and R. Lindh. Theoretical study of the internal charge transfer in aminobenzonitriles. *J. Am. Chem. Soc.*, 117:3189, 1995.

[65] M. Merchán, R. Pou-Amérigo, and B. O. Roos. A theoretical study of the dissociation energy of Ni_2^+. A case of broken symmetry. *Chem. Phys. Letters*, 252:405, 1996.

[66] M. P. Fülscher, S. Matzinger, and B. Bally. Excited states in polyene radical cations. An ab initio theoretical study. *Chem. Phys. Letters*, 236:167, 1995.

[67] M. Rubio, M. Merchán, E. Ortí, and B. O. Roos. Theoretical study of the electronic spectra of the biphenyl cation and anion. *J. Phys. Chem.*, 99:14980, 1995.

[68] V. Barone and M. Cossi. *J. Phys. Chem. A*, 102:1995, 1998.

[69] M. Cossi, N. Rega, G. Scalmani, and V. Barone. *J. Chem. Phys.*, 114:5691–5701, 2001.

[70] G. Karlström. A new approach to the modeling of dielectric media effects in ab initio quantum chemical calculations. *J. Phys. Chem.*, 92:1315–1318, 1988.

[71] L. Serrano-Andrés, M. P. Fülscher, and G. Karlström. *Int. J. Quantum Chem.*, 65:167, 1997.

[72] J. Tomasi and M. Persico. *Chem. Rev.*, 94:2027, 1994.

[73] M. Cossi and V. Barone. *J. Chem. Phys.*, 112:2427, 2000.

[74] A. Bernhardsson, R. Lindh, G. Karlström, and B. O. Roos. Direct self-consistent reaction field with Pauli repulsion: Solvation effects on methylene peroxide. *Chem. Phys. Letters*, 251:141, 1996.

[75] W. F. Forbes and R. Shilton. *J. Am. Chem. Soc.*, 81:786, 1959.

[76] N. Douglas and N. M. Kroll. *Ann.Phys.*, 82:89, 1974.

[77] B.A. Hess. *Phys.Rev. A*, 33:3742, 1986.

[78] P.-Å. Malmqvist, B. O. Roos, and B. Schimmelpfennig. The restricted active space (ras) state interaction approach with spin-orbit coupling. *Chem. Phys. Letters*, 357:230, 2002.

[79] B. A. Hess, C.M. Marian, U. Wahlgren, and O. Gropen. *Chem. Phys. Letters*, 251:365, 1996.

[80] B. Schimmelpfennig. Amfi, an atomic mean-field spin-orbit integral program. Stockholm University, 1996.

[81] B. O. Roos and P.-Å. Malmqvist. On the effects of spin-orbit coupling on molecular properties: Dipole moment and polarizability of pbo and spectroscopic constants for the ground and excited states. *Adv. Quant. Chem.*, 47:37, 2004.

[82] R. Lindh. The reduced multiplication scheme of the Rys-Gauss quadrature for 1st order integral derivatives. *Theor. Chim. Acta*, 85:423–440, 1993.

[83] K. Andersson and B. O. Roos. Multiconfigurational second-order perturbation theory: A test of geometries and binding energies. *Int. J. Quantum Chem.*, 45:591, 1993.

[84] G. Ghigo, B. O. Roos, and P.-Å. Malmqvist. A modified definition of the zeroth order hamiltonian in multiconfigurational perturbation theory (caspt2). *Chem. Phys. Letters*, 396:142, 2004.

[85] T. Thorsteinsson, D. L. Cooper, J. Gerratt, P. B. Karadakov, and M. Raimondi. Modern valence bond representations of casscf wavefunctions. *Theor. Chim. Acta*, 93:343, 1996.

[86] D. L. Cooper, T. Thorsteinsson, and J. Gerratt. Fully-variational optimization of modern vb wave functions using the casvb strategy. *Int. J. Quantum Chem.*, 65:439, 1997.

[87] D. L. Cooper, T. Thorsteinsson, and J. Gerratt. Modern vb representations of casscf wave functions and the fully-variational optimization of modern vb wave functions using the casvb strategy. *Adv. Quant. Chem.*, 32:51, 1998.

[88] T. Thorsteinsson and D. L. Cooper. An overview of the casvb approach to modern valence bond calculations. page 303, 2000.

[89] P. Neogrády, M. Urban, and I. Hubač. *J. Chem. Phys.*, 100:3706, 1994.

[90] P. Neogrády, M. Urban, and I. Hubač. *J. Chem. Phys.*, 97:5074, 1992.

[91] P. J. Knowles, C. Hampel, and H. J. Werner. *J. Chem. Phys.*, 99:5219, 1993.

[92] M. Urban, J. Noga, S. J. Cole, and R. J. Bartlett. *J. Chem. Phys.*, 83:4041, 1985.

[93] K. Raghavachari, G. W. Trucks, J. A. Pople, and M. Head-Gordon. *Chem. Phys. Letters*, 157:479, 1989.

[94] R. Ahlrichs, P. Scharf, and C Ehrhardt. The coupled pair functional (cpf). a size consistent modification of the ci(sd) based on an energy functional. *J. Chem. Phys.*, 82:890, 1985.

[95] D. P. Chong and S. R. Langhoff. A modified coupled-pair functional approach. *J. Chem. Phys.*, 84:5606, 1986.

[96] R. J. Gdanitz and R. Ahlrichs. The averaged coupled-pair functional (acpf): a size-extensive modification of mr ci(sd). *Chem. Phys. Letters*, 143:413, 1988.

[97] B. Roos. A new method for large-scale CI calculations. *Chem. Phys. Letters*, 15:153–157, 1972.

[98] I. Shavitt. *Int. J. Quantum Chem.*, S11:131, 1977.

[99] P. E. M. Siegbahn. *J. Chem. Phys.*, 72:1647, 1980.

[100] J. M. del Campo and A. M. Köster. A hierarchical transition state search algorithm. *J. Chem. Phys.*, 129:Art. Nr. 024107, 2008.

[101] R. C. Raffenetti. *J. Chem. Phys.*, 58:4452, 1973.

[102] J. Almlöf and P. R. Taylor. General contraction of gaussian basis sets. I. Atomic natural orbitals for first- and second row atoms. *J. Chem. Phys.*, 86:4070, 1986.

[103] P.-O. Widmark, P.-Å. Malmqvist, and B. O. Roos. Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions. I. First row atoms. *Theor. Chim. Acta*, 77:291, 1990.

[104] P.-O. Widmark, B. J. Persson, and B. O. Roos. Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions. II. Second row atoms. *Theor. Chim. Acta*, 79:419, 1991.

[105] R. Pou-Amérigo, M. Merchán, I. Nebot-Gil, P.-O. Widmark, and B. O. Roos. Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions III. First row transition metal atoms. *Theor. Chim. Acta*, 92:149–181, 1995.

[106] K. Pierloot, B. Dumez, P.-O. Widmark, and B. O. Roos. Density matrix averaged atomic narural orbital (ANO) basis sets for correlated molecular wave functions. IV. Medium size basis sets for the atoms H–Kr. *Theor. Chim. Acta*, 90:87, 1995.

[107] J. Pipek and P.G. Mezey. *J. Chem. Phys.*, 90:4916, 1989.

[108] S. F. Boys. *Rev. Mod. Phys.*, 32:296, 1960.

[109] J. M. Foster and S. F. Boys. *Rev. Mod. Phys.*, 32:300, 1960.

[110] C. Edmiston and K. Ruedenberg. *Rev. Mod. Phys.*, 35:457, 1963.

[111] F. Aquilante and T. B. Pedersen and A. Sánches de Merás and H. Koch. Fast noniterative orbital localization for large molecules. *J. Chem. Phys.*, 125:174101, 2006.

[112] J. E. Subotnik and Y. Shao and W. Z. Liang and M. Head-Gordon. *J. Chem. Phys.*, 121:9220, 2004.

[113] L. Gagliardi, R. Lindh, and G. Karlström. Local properties of quantum chemical systems: the loprop approach. *J. Chem. Phys.*, 121:4494, 2004.

[114] A. D. Becke and E. R. Johnson. Exchange-hole dipole moment and the dispersion interaction. *J. Chem. Phys.*, 122:154104, 2005.

[115] A. Bernhardsson, R. Lindh, J. Olsen, and M. Fülscher. A direct implementation of the second-order derivatives of multiconfigurational SCF energies and an analysis of the preconditioning the associated response equation. *Mol. Phys.*, 96:617–628, 1999.

[116] J. Stålring, A. Bernhardsson, and R. Lindh. Analytical gradients of a state average MCSCF state and a state average diagnostic. *Mol. Phys.*, 99:103–114, 2001.

[117] J. Olsen, B. O. Roos, P. Jørgensen, and H. J. Aa. Jensen. *J. Chem. Phys.*, 89:2185, 1985.

[118] P.-Å. Malmqvist, A. Rendell, and B. O. Roos. *J. Phys. Chem.*, 94:5477, 1990.

[119] B. O. Roos, P. R. Taylor, and P. E. M. Siegbahn. A Complete Active Space SCF method (CASSCF) using a density-matrix formulated super-CI approach. *Chem. Phys.*, 48:157–173, 1980.

[120] B. O. Roos. The Complete Active Space SCF method in a fock-matrix-based super-CI formulation. *Int. J. Quantum Chem.*, S14:175–189, 1980.

[121] F. Aquilante and T. B. Pedersen and R. Lindh. Low-cost evaluation of the exchange fock matrix from cholesky and density fitting representations of the electron repulsion integrals. *J. Chem. Phys.*, 126:194106, 2007.

[122] P.-Å. Malmqvist. Calculation of transition density matrices by non-unitary orbital transformations. *Int. J. Quantum Chem.*, 30:479, 1986.

[123] P.-Å. Malmqvist and B. O. Roos. The CASSCF state interaction method. *Chem. Phys. Letters*, 155:189, 1989.

[124] J. Almlöf, Jr. K. Faegri, and K. Korsell. *J. Comput. Chem.*, 3:385, 1982.

[125] D. Cremer and J. Gauss. *J. Comput. Chem.*, 7:274, 1986.

[126] M. Häser and R. Ahlrichs. *J. Comput. Chem.*, 10:104, 1989.

[127] G. Karlström. *Chem. Phys. Letters*, 67:348, 1979.

[128] G. Karlström. *Int. J. Quantum Chem.*, 45:31, 1993.

[129] T.H. Fischer and J. Almlöf. *J. Phys. Chem.*, 96:9768, 1992.

[130] S. H. Vosko, L. Wilk, and M. Nusair. *Canadian J. Phys.*, 58:1200, 1980.

[131] A. D. Becke. *Phys. Rev. A*, 38:3098, 1988.

[132] P. Hohenberg and W. Kohn. *Phys. Rev.*, 136:B864, 1964.

[133] W. Kohn and L. J. Sham. *Phys. Rev.*, 140:A1133, 1965.

[134] J. C. Slater. *Quantum Theory of Molecular and Solids. Vol. 4: The Self-Consistent Field for Molecular and Solids.* McGraw-Hill, New York, 1974.

[135] C. Lee, W. Yang, and R. G. Parr. *Phys. Rev. B*, 37:785, 1988.

[136] B. Miehlich, A. Savin, H. Stoll, and H. Preuss. *Chem. Phys. Lett.*, 157:200, 1989.

[137] A. D. Becke. *J. Chem. Phys.*, 98:5648, 1993.

[138] J.P. Perdew, K. Burke, and M. Ernzerhof. Generalized gradient approximation made simple. *Phys. Rev. Letters*, 77:3865–3868, 1996.

[139] J.P. Perdew, M. Ernzerhof, and K. Burke. Rationale for mixing exact exchange with density functional approximations. *J. Chem. Phys.*, 105:9982–9985, 1996.

[140] Y. Zhao and D. G. Truhlar. *J. Chem. Phys.*, 125:194101, 2006.

[141] Y. Zhao and D. G. Truhlar. 110:13126, 2006.

[142] Y. Zhao and D. G. Truhlar. *Theor. Chim. Acta*, 120:215, 2008.

[143] Y. Zhao and D. G. Truhlar. *Acc. Chem. Res.*, 41:157, 2008.

[144] R. Lindh, U. Ryu, and B. Liu. The reduced multiplication scheme of the Rys quadrature and new recurrence relations for auxiliary function based two-electron integral evaluation. *J. Chem. Phys.*, 95:5889–5897, 1991.

[145] E.R. Davidson. Use of double cosets in constructing integrals over symmetry orbitals. *J. Chem. Phys.*, 62:400, 1975.

[146] B. G. Johnson, P. M. W. Gill, and J. A. Pople. The performance of a family of density functional methods. *J. Chem. Phys.*, 98:5612, 1993.

[147] N. C. Handy, D. J. Tozer, G. J. Laming, C. W. Murray, and R. D. Amos. Analytic second derivatives of the potential energy surface. *Israel J. Chem.*, 33:331, 1993.

[148] J. Baker, J. Andzelm, and A. Scheiner. The effect of grid quality and weight derivatives in density functional calculations. *J. Chem. Phys.*, 101:8894, 1994.

[149] M. Reiher and A. Wolf. Exact decoupling of the dirac hamiltonian. i. general theory. *J. Chem. Phys.*, 121:2037–2047, 2004.

[150] M. Reiher and A. Wolf. Exact decoupling of the dirac hamiltonian. ii. the generalized douglas-kroll-hess transformation up to arbitrary order. *J. Chem. Phys.*, 121:10945–10956, 2004.

[151] M. Reiher. Douglas-kroll-hess theory: a relativistic electrons-only theory for chemistry. *Theor. Chim. Acta*, 116:241–252, 2006.

[152] A. Wolf, M. Reiher, and B. Hess. The generalized douglas-kroll transformation. *J. Chem. Phys.*, 117:9215–9226, 2002.

[153] A. Wolf and M. Reiher. Exact decoupling of the dirac hamiltonian. iii. molecular properties. *J. Chem. Phys.*, 124:Art. Nr. 06102, 2006.

[154] A. Wolf and M. Reiher. Exact decoupling of the dirac hamiltonian. iv. automated evaluation of molecular properties within the douglas-kroll-hess theory up to arbitrary order. *J. Chem. Phys.*, 124:Art. Nr. 06103, 2006.

[155] R. Lindh, A. Bernhardsson, G. Karlström, and P.-Å. Malmqvist. On the use of a Hessian model function in molecular geometry optimizations. *Chem. Phys. Letters*, 241:423–428, 1995.

[156] C. Peng, P. Y. Ayala, H. B. Schlegel, and M. J. Frisch. Using redundant internal coordinates to optimize equilibrium geometries and transition states. *J. Comput. Chem.*, 17:49, 1996.

[157] P. Pulay and G. Fogarasi. Geometry optimization in redundant internal coordinates. *J. Chem. Phys.*, 96:2856, 1992.

[158] J. Baker, A. Kessi, and B. Delley. The generation and use of localized internal coordinates in geometry optimization. *J. Chem. Phys.*, 105:192, 1996.

[159] R. Lindh, A. Bernhardsson, and M. Schütz. Force-constant weighted redundant coordinates in molecular geometry optimizations. *Chem. Phys. Letters*, 303:567–575, 1999.

[160] J. Baker. Techniques for geometry optimization: A comparison of cartesian and natural internal coordinates. *J. Comput. Chem.*, 14:1085, 1993.

[161] M. J. Frisch, G. W. Trucks, H. B. Schlegel, P. M. W. Gill, B. G. Johnson, M. A. Robb, J. R. Cheeseman, T. A. Keith, G. A. Petersson, J. A. Montgomery, K. Raghavachari, M. A. Al-Laham, V. G. Zakrzewski, J. V. Ortiz, J. B. Forseman, J. Cioslowski, B. B. Stefanov, A. Nanayakkara, M. Challacombe, C. Y. Peng, P. Y. Ayala, W. Chen, M. W. Wong, J. L. Andres, E. S. Replogle, R. Gomperts, R. L. Martin, D. J. Fox, J. S. Binkley, D. J. Defrees, J. Baker, J. J. P. Stewart, M. Head-Gordon, C. Gonzalez, and J. A. Pople. *Gaussian 94 (Revision A.1)*, 1995.

[162] M.J.D. Powell. Recent advances in unconstrained optimization. mathematical programming. *Math. Prog.*, 1:26, 1971.

[163] R. Fletcher. *Practical Methods of optimization.* Wiley, New York, 1981.

[164] C.G. Broyden. The convergence of a class of double-rank minimization algorithms. the new algorithm. *J. Inst. Math. Appl.*, 6:222, 1970.

[165] R. Fletcher. A new approach to variable metric algorithms. *Comp. J.*, 13:317, 1970.

[166] D. Goldfarb. A family of variable-metric methods derived by variational means. *Math. Comp.*, 24:23, 1970.

[167] D.F. Shanno. Conditioning of quasi-newtonian methods for function minimization. *Math. Comp.*, 24:647, 1970.

[168] J.E. Dennis and R.R. Schnabel. Least change secant updates for quasi-newtonian methods. *SIAM Rev.*, 21:443, 1979.

[169] J. M. Bofill. Updated hessian matrix and the restricted step method for locating transition structures. *J. Comput. Chem.*, 15:1, 1994.

[170] A. Banerjee, N. Adams, J. Simons, and R. Shepard. Search for stationary points on surfaces. *J. Phys. Chem.*, 89:52, 1985.

[171] E. Besalú and J. M. Bofill. On the automatic restricted-step rational-function-optimization method. *Theor. Chim. Acta*, 100:265, 1998.

[172] P. Császár and P. Pulay. Geometry optimization by direct inversion in the iterative subspace. *J. of Mol. Struct.*, 114:31, 1984.

[173] P. Pulay. Convergence acceleration of iterative sequences. the case of scf iterations. *Chem. Phys. Lett.*, 73:393, 1980.

[174] P. Pulay. Improved scf convergence acceleration. *J. Comp. Chem.*, 3:556, 1982.

[175] H. Sellers. The c^2-diis convergence acceleration algorithm. *Int. J. Quantum Chem.*, 45:31, 1993.

[176] C. J.Cerjan and W. H. Miller. On finding transition states. *J. Chem. Phys.*, 75:2800, 1981.

[177] H. B. Schlegel. Optimization of equilibrium geometries and transition structures. In K. P. Lawley, editor, *Advances in Chemical Physics; Ab Initio Methods in Quantum Chemistry - I*, chapter 69, page 249. John Wiley & Sons Ltd., Chichester, England, 1987.

[178] A.J. Sadlej. Medium-size polarized basis sets for high-level correlated calculations of molecular electric properties. *Collect. Czech. Chem. Commun.*, 53:1995, 1988.

[179] A. J. Sadlej. Medium-size polarized basis sets for high-level-correlated calculations of molecular electric properties. II. Second-row atoms: Si through Cl. *Theor. Chim. Acta*, 79:123, 1991.

[180] A. J. Sadlej and M. Urban. Medium-size polarized basis sets for high-level-correlated calculations of molecular electric properties. III. Alkali (Li. Na, K, Rb) and alkaline-earth (Be, Mg, Ca, Sr). *J. Mol. Struc.*, 234:147, 1991.

[181] A. J. Sadlej. Medium-sized polarized basis sets for high-level-correlated calculations of molecular electric properties. IV. Third-row atoms: Ge through Br. *Theor. Chim. Acta*, 81:45, 1991.

[182] A. J. Sadlej. Medium-size polarized basis sets for high-level-correlated calculations of molecular electric properties. V. Fourth-row atoms: Sn through I. *Theor. Chim. Acta*, 81:339, 1992.

[183] V. Kellö and A. J. Sadlej. Estimates of relativistic contributions to molecular properties. *J. Chem. Phys.*, 93:8122, 1990.

[184] A. J. Sadlej and M. Urban. Mutual dependence of relativistic and electron correlation contributions of molecular properties: The dipole moment of AgH. *Chem. Phys. Letters*, 176:293, 1991.

[185] S. Huzinaga, L. Seijo, Z. Barandiarán, and M. Klobukowski. The ab initio model potential method. main group elements. *J. Chem. Phys.*, 86:2132, 1987.

[186] Z. Barandiarán and L. Seijo. The ab initio model potential representation of the crystalline environment. theoretical study at the local distortion on nacl:cu$^+$. *J. Chem. Phys.*, 89:5739–5746, 1988.

[187] Z. Barandiarán, L. Seijo, and S. Huzinaga. The ab initio model potential method. second series transition metals. *J. Chem. Phys.*, 93:5843, 1990.

[188] C. Wittborn and U. Wahlgren. New relativistic effective core potentials for heavy elements. *Chem. Phys.*, 201:357, 1995.

[189] F. Rakowitz, C. M. Marian, L. Seijo, and U. Wahlgren. Spin-free relativistic no-pair *ab initio* core model potentials and valence basis sets for the transition metal elements Sc to Hg. Part I. *J. Chem. Phys.*, 110:3678, 1999.

[190] F. Rakowitz, C. M. Marian, and L. Seijo. Spin-free relativistic no-pair *ab initio* core model potentials and valence basis sets for the transition metal elements Sc to Hg. Part II. *J. Chem. Phys.*, 111:10436, 1999.

[191] Z. Barandiarán and L. Seijo. In S. Fraga, editor, *Computational Chemistry: Structure, Interactions and Reactivity*, volume 77B of *Studies in Physical and Theoretical Chemistry*, pages 435–461. Elsevier, Amsterdam, 1992.

[192] J. C. Phillips and L. Kleinman. New method for calculating wave functions in crystal and molecules. *Phys. Rev.*, 116:287, 1959.

[193] S. Huzinaga and A. A. Cantu. J. Chem. Phys. **55**, 5543 (1971); S. Huzinaga, D. McWilliams, and A. A. Cantu, Adv. Quantum Chem. **7**, 187 (1973).

[194] J. L. Pascual, L. Seijo, and Z. Barandiarán. Ab initio model potential study of environmental effects on the jahn-teller parameters of cu^{2+} and ag^{2+} impurities in mgo, cao, and sro hosts. *J. Chem. Phys.*, 98:9715, 1993.

[195] M. Pelissier, N. Komiha, and J.-P. Daudey. *J. Comp. Chem.*, 9:298, 1988.

[196] P. J. Hay and W. R. Wadt. Ab initio effective core potentials for molecular calculations. potentials for the transition metal atoms sc-hg. *J. Chem. Phys.*, 82:270, 1985.

[197] P. J. Hay and W. R. Wadt. Ab initio effective core potentials for molecular calculations. potentials for main group elements na-bi. *J. Chem. Phys.*, 82:284, 1985.

[198] P. J. Hay and W. R. Wadt. Ab initio effective core potentials for molecular calculations. potentials for k to au including the outermost core orbitals. *J. Chem. Phys.*, 82:299, 1985.

[199] P. Fuentealba, H. Preuss, H. Stoll, and L. v. Szentpaly. *Chem. Phys. Lett.*, 89:418, 1982.

[200] P. Fuentealba, L. v. Szentpaly, H. Preuss, and H. Stoll. *J. Phys. B*, 18:1287, 1985.

[201] G. Igel-Mann, H. Stoll, and H. Preuss. *Mol. Phys.*, 65:1321, 1988.

[202] A. Bergner, M. Dolg, W. Kuechle, H. Stoll, and H. Preuss. *Mol. Phys.*, 80:1431, 1993.

[203] P. Fuentealba, H. Stoll, L. v. Szentpaly, P. Schwerdtfeger, and H. Preuss. *J. Phys. B*, 16:1323, 1983.

[204] M. Kaupp, P. v. R. Schleyer, H. Stoll, and H. Preuss. *J. Chem. Phys.*, 94:1360, 1991.

[205] M. Dolg, U. Wedig, H. Stoll, and H. Preuss. *J. Chem. Phys.*, 86:866, 1987.

[206] U. Wedig, M. Dolg, H. Stoll, and H. Preuss. In A. Veillard, editor, *Quantum Chemistry: The Challenge of Transition Metals and Coordination Chemistry*, page 79. Reidel, Dordrecht, 1986.

[207] L. v. Szentpaly, P. Fuentealba, H. Preuss, and H. Stoll. *Chem. Phys. Lett.*, 93:555, 1982.

[208] D. Andrae, U. Haeussermann, M. Dolg, H. Stoll, and H. Preuss. *Theor. Chim. Acta*, 77:123, 1990.

[209] H. Stoll, P. Fuentealba, P. Schwerdtfeger, J. Flad, L. v. Szentpaly, and H. Preuss. *J. Chem. Phys.*, 81:2732, 1984.

[210] W. Kuechle, M. Dolg, H. Stoll, and H. Preuss. *Mol. Phys.*, 74:1245, 1991.

[211] G. Igel-Mann. Thesis, Univeristy of Stuttgart, Germany, 1987.

[212] M. Dolg, H. Stoll, and H. Preuss. *Theor. Chim. Acta*, 85:441, 1993.

[213] M. Dolg, H. Stoll, and H. Preuss. *J. Chem. Phys.*, 90:1730, 1989.

[214] M. Dolg, P. Fulde, W. Kuechle, C.-S. Neumann, and H. Stoll. *J. Chem. Phys.*, 94:3011, 1991.

[215] M. Dolg, H. Stoll, A. Savin, and H. Preuss. *Theor. Chim. Acta*, 75:173, 1989.

[216] M. Dolg, H. Stoll, H.-J. Flad, and H. Preuss. *J. Chem. Phys.*, 97:1162, 1992.

[217] M. Dolg, H. Stoll, H. Preuss, and R.M. Pitzer. *J. Phys. Chem.*, 97:5852, 1993.

[218] P. Schwerdtfeger, M. Dolg, W.H.E. Schwarz, G.A. Bowmaker, and P.D.W. Boyd. *J. Chem. Phys.*, 91:1762, 1989.

[219] U. Haeussermann, M. Dolg, H. Stoll, and H. Preuss. *Mol. Phys.*, 78:1211, 1993.

[220] W. Kuechle, M. Dolg, H. Stoll, and H. Preuss. *J. Chem. Phys.*, 100:7535, 1994.

[221] A. Nicklass, M. Dolg, H. Stoll, and H. Preuss. *J. Chem. Phys.*, 102, 1995.

[222] T. Leininger, A. Nicklass, H. Stoll, M. Dolg, and P. Schwerdtfeger. *J. Chem. Phys.*, 105:1052, 1996.

[223] X. Cao, M. Dolg, and H. Stoll. *J. Chem. Phys.*, 118:487, 2003.

[224] L. R. Kahn and W. A. Goddard. *Chem. Phys. Letters*, 2:667, 1968.

[225] P. A. Christiansen, Y. S. Lee, and K. S. Pitzer. *J. Chem. Phys.*, 71:4445, 1979.

[226] P. Durand and J. C. Barthelat. *Theor. Chim. Acta*, 38:283, 1975.

[227] C.-K. Skylaris, L. Gagliardi, N. C. Handy, A. G. Ioannou, S. Spencer, A. Willetts, and A. M. Simper. An efficient method for calculating effective core potential integrals which involve projection operators. *Chem. Phys. Letters*, 296:445–451, 1998.

[228] I. Shavitt. *Int. J. Quantum Chem.*, S12:5, 1978.

[229] L. Serrano-Andrés, M. Merchán, M. P. Fülscher, and B. O. Roos. A theoretical study of the electronic spectrum of thiophene. *Chem. Phys. Letters*, 211:125, 1993.

[230] A. Öhrn. *Development and Application of a First Principle Molecular Model for Solvent Effects.* PhD thesis, University of Lund, Theor. Chemistry, Chem. Center, P.O.B. 124,S-221 00 Lund, Sweden, 2007.

[231] M. Schütz and R. Lindh. *Theor. Chim. Acta*, 95:13, 1997.

[232] P. E. M. Siegbahn, J. Almlöf, J. Heiberg, and B. O. Roos. The Complete Active Space SCF (CASSCF) method in a Newton-Raphson formulation with application to the HNO molecule. *J. Chem. Phys.*, 74:2384–2396, 1981.

[233] T. A. Halgren and W. N. Lipscomb. *Chem. Phys. Letters*, 50:225, 1977.

[234] W. J. Hehre. *Practical Strategies for Electronic Structure Calculations.* Wavefunction, Irvine, California, 1995.

[235] K. Pierloot, B. Dumez, P.-O. Widmark, and B. O. Roos. Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions. IV. Medium size basis sets for the atoms H–Kr. *Theor. Chim. Acta*, 90:87, 1995.

[236] K. Andersson, M. Baryz, A. Bernhardsson, M. R. A. Blomberg, P. Boussard, D. L. Cooper, T. Fleig, M. P. Fülscher, B. Hess, G. Karlström, R. Lindh, P.-Å. Malmqvist, P. Neogrády, J. Olsen, B. O. Roos, A. J. Sadlej, B. Schimmelpfennig, M. Schütz, L. Seijo, L. Serrano, P. E. Siegbahn, J. Stålring, T. Thorsteinsson, V. Veryazov, Ulf Wahlgren, and P.-O. Widmark. *MOLCAS Version 5.0.* Dept. of Theor. Chem., Chem. Center, Univ. of Lund, P.O.B. 124, S-221 00 Lund, Sweden, Lund, 2000.

[237] R. Lindh, P.-Å. Malmqvist, and L. Gagliardi. *Theor. Chem. Acc.*, 106:178, 2001.

[238] O. Christiansen, J.Gauss, and B. Schimmelpfennig. *Phys. Chem. Chem. Phys.*, 2:965, 2000.

[239] U. Ryde, M. H. M. Olsson, K. Pierloot, and B. O. Roos. The cupric geometry of blue copper proteins is not strained. *J. Mol. Biol.*, 261:586–596, 1996.

[240] U. Ryde, M. H. M. Olsson, B. O. Roos, J. O. A. De Kerpel, and K. Pierloot. On the role of strain in blue copper proteins. *J. Biol. Inorg. Chem.*, 5:565–574, 2000.

[241] U. Ryde and M. H. M. Olsson. Structure, strain, and reorganisation energy of blue-copper models in the protein. *Int. J. Quantum Chem.*, 81:335–347, 2001.

[242] N. W. Moriarty and G. Karlström. *J. Phys. Chem.*, 100:17791, 1996.

[243] A. Öhrn and G. Karlström. *Mol. Phys.*, 104:3087, 2006.

[244] A. Öhrn and F. Aquilante. *Phys. Chem. Chem. Phys.*, 9:470, 2007.

[245] O. Engkvist, P.-O. Åstrand, and G. Karlström. Accurate intermolecular potentials obtained from molecular wave funcions: Bridging the gap between quantum chemistry and molecular simulations. *Chem. Rev.*, 100:4087–4108, 2000.

[246] A. Wallqvist, P. Ahlström, and G. Karlström. *J. Phys. Chem.*, 94:1649–1656, 1990.

Index