DALTON Release 2 Program Manual

C. Angeli, K. L. Bak, V. Bakken, O. Christiansen, R. Cimiraglia,
S. Coriani, P. Dahle, E. K. Dalskov, T. Enevoldsen, B. Fernandez,
C. Hättig, K. Hald, A. Halkier, H. Heiberg, T. Helgaker, H. Hettema,
H. J. Aa. Jensen, D. Jonsson, P. Jørgensen, S. Kirpekar, W. Klopper,
R. Kobayashi, H. Koch, A. Ligabue, O. B. Lutnæs, K. V. Mikkelsen, P. Norman,
J. Olsen, M. J. Packer, T. B. Pedersen, Z. Rinkevicius, E. Rudberg,
T. A. Ruden, K. Ruud, P. Sałek, A. Sanchez de Meras, T. Saue,
S. P. A. Sauer, B. Schimmelpfennig, K. O. Sylvester-Hvid, P. R. Taylor,
O. Vahtras, D. J. Wilson, and H. Ågren,

Contents

Preface

This is the documentation for the DALTON quantum chemistry program — Release 2.0 Rev.0, March 2005 — for computing Hartree–Fock, Kohn–Sham DFT, MCSCF, NEVPT2, MP2, and coupled cluster wave functions, including here MP2-R12 energies, and for calculating molecular properties and potential energy surfaces.

We emphasize here the conditions under which the program is distributed. It is furnished for your own use, and you may not redistribute it further, either in whole or in part. Any one interested in obtaining DALTON should check out the DALTON homepage at `http://www.kjemi.uio.no/software/dalton/dalton.html`.

Any use of the program that results in published material should cite the following:

"DALTON, a molecular electronic structure program, Release 2.0 (2005), see http://www.kjemi.uio.no/software/dalton/dalton.html"

The program has been developed over a large number of years and include an extensive list of contributing authors:

C. Angeli, K. L. Bak, V. Bakken, O. Christiansen, R. Cimiraglia, S. Coriani,
P. Dahle, E. K. Dalskov, T. Enevoldsen, B. Fernandez, C. Hättig, K. Hald,
A. Halkier, H. Heiberg, T. Helgaker, H. Hettema, H. J. Aa. Jensen, D. Jonsson,
P. Jørgensen, S. Kirpekar, W. Klopper, R. Kobayashi, H. Koch, A. Ligabue,
O. B. Lutnæs, K. V. Mikkelsen, P. Norman, J. Olsen, M. J. Packer,
T. B. Pedersen, Z. Rinkevicius, E. Rudberg, T. A. Ruden, K. Ruud, P. Sałek,
A. Sanchez de Meras, T. Saue, S. P. A. Sauer, B. Schimmelpfennig,
K. O. Sylvester-Hvid, P. R. Taylor, O. Vahtras, D. J. Wilson, and H. Ågren.

The program represents experimental code that is under constant development. No guarantees of any kind are provided, and the authors accept no responsibility for the performance of the code or for the correctness of the results.

Chapter 1

Introduction

The DALTON program system is designed to allow convenient, automated determination of a large number of molecular properties based on an HF, DFT, MP2, coupled cluster, or MCSCF reference wave function. Additions to the Dalton 2.0 release include density functional theory implemented fully up to quadratic response theory for closed-shell systems, as well as second derivatives for properties involving perturbation-dependent basis sets such a molecular Hessians, magnetizabilities and nuclear shieldings. Also available is the NEVPT2 approach (the n-electron valence second-order perturbation theory) for calculating accurate energetics of multireference-based systems including also dynamic electron correlation, and highly accurate MP2-R12 methods are available for high-accuracy energetics of single-reference dominated closed-shell systems.

The program consists of seven separate components, developed more or less independently. HERMIT is the integral generator, generating ordinary atomic and molecular integrals appearing in the time-independent, non-relativistic Schrödinger equation, as well as an extensive number of integrals related to different molecular properties. ERI is a vectorized and distribution-oriented integral generator that may be invoked in certain calculations, in particular in integral-direct coupled cluster calculations. SIRIUS is the (MC)SCF wave function optimization part, and is described comprehensively in the METECC-94 book [1]. ABACUS evaluates the second-order molecular properties of interest for SCF and MCSCF wave functions as well density-functional theory, in particular second-order static molecular properties in which the basis set depends on the applied perturbation. RESPONSE is a general-purpose program for evaluating response functions, up to cubic response functions for (MC)SCF wave functions, quadratic response for DFT, and linear response for the Second-Order Polarization Propagator Approximation (SOPPA) and Second-Order Polarization Propagator Approximation with Coupled Cluster Singles and Doubles Amplitudes - SOPPA(CCSD). A new density functional theory module has been added in the 2.0 release of the program. Existing modules have been extended with the NEVPT2 and the MP2-R12

methodologies.

Throughout this manual there will be references to articles describing the implementation of a specific molecular property or input option. This will hopefully suffice to give the reader a proper theoretical understanding of the current implementation. Only one reference otherwise not mentioned in the text, the treatment of symmetry, is given here [2].

DALTON is in many respects an "expert's" program. This is most noticeable in the range and selection of molecular properties that may be calculated and the flexibility and stability of the wave function implementations available. As described in the individual sections, the range of molecular properties, some of which are highly non-standard, is fairly large. On the other hand, several common properties can perhaps more easily be calculated using other, (non-)commercially available, quantum chemistry program packages.

We have tried when writing this manual to emphasize the modularity of the input, as well as indicating the advantages that may be obtained from the flexibility of the input. Yet, whether the authors of this manual have succeeded or not, is up to the reader to decide, and any comments and suggestions for improvements, both on the manual as well as the program, will be much welcome.

1.1 General description of the manual

The manual is divided into three main sections:

The Installation Guide (Chapter 3—4) describes the installation and configuration of the DALTON program. Although of interest primarily to those installing the program, the section describes the program's use of memory and the default settings for various parameters in the program and how they may be changed. Thus, we strongly recommend that all users read through this section at least once.

The User's Guide (Chapter 5—20) will be the most important part of this manual for most users. These chapters provide a short introduction on how to do a calculation with the DALTON program, considering both the supplied job run script `dalton` as well as some examples of input files for a complete run of the program. A short description of the different output files is also given.

These chapters also describe the input file needed to calculate different molecular properties. Some suggestions for how to do a calculation most efficiently, as well as recommendations for basis sets to be used when calculating non-standard molecular properties are also included. Although highly biased, we still hope that they may be of some use to the user. These are the most useful sections for ordinary calculations done with the DALTON program.

> *In these chapters there are boxes indicating what we call* reference literature. *This reference literature not only serve as the best introduction to the implementational aspects of a given calculational procedure, but is also our recommended reference in published work employing specific parts of the code.*

The Reference Manual (Chapter 21—27) will for most calculations be of little use to the ordinary user apart from Chapter 23, which describes the most important file in any quantum chemistry calculation: the description of the molecule being studied. However, all possible keywords that may be given to direct the calculation using DALTON is documented in this Reference Manual.

Appendix A gives a description of various tools provided to us from different users for pre- or post-processing of DALTON input/output.

Before starting we thus recommend reading the following chapters: 5, 6, and 23, and one or more of chapters 7—20, depending on the calculations you would like to do with the program. Chapters 3 and 4 are also highly recommended.

Questions and bug reports should be directed to `dalton-admin@kjemi.uio.no`. Help on input and/or error messages produced during execution of the program should be directed to the DALTON mailing list, `dalton-users@kjemi.uio.no`.

1.2 Acknowledgments

We are grateful to Prof. Björn O. Roos and Dr. Per-Olof Widmark, University of Lund, for kindly allowing us to include in the distribution of the DALTON program routines for integral presorting (BOR), two-electron integral transformation routines (BOR) and routines for synchronous and asynchronous I/O (POW).

We are grateful to professor Knut Færgri (University of Oslo) who took the time to reoptimize the basis set of van Duijneveldt in order to provide in electronic form a basis set of similar performance to these widely used basis set, included here as the Not Quite van Duijneveldt (NQvD) basis sets.

We would also like to express our gratitude to Andreas Hesselmann (University of Düsseldorf), Gilbert Hangartner (University of Freiburg), and Antonio Rizzo (Istituto di Chimica Quantistica ed Energetica Molecolare del CNR, Pisa) for allowing us to distribute their utility programs for pre- and post-processing of DALTON input/output-files, as described in the Appendix.

A long list of users, too long to mention here, is thanked for an enormous amount of valuable feedback on the performance of the code. Without their assistance, the code would not have looked the way it does now.

Chapter 2

New features in Dalton

2.1   New features in Dalton 2.0

There are several new additions introduced in the Dalton 2.0. The most important and largest extension to the code is the addition of a complete Density Functional Theory (DFT) module, including up to quadratic response properties as well as an extensive open-shell (spin-restricted) module. The main new features and changes are summarized below:

1. **Density Functional Theory:**

   The largest new extension to Dalton 2.0 is the addition of a complete Density Functional Theory (DFT) module. It contains 24 different exchange-correlation functions, and is implemented for energies, linear and quadratic response functions (for both singlet and triplet perturbing operators) and geometric Hessians, nuclear shielding tensors, and magnizabilities as well as indirect spin–spin coupling constants. Energies and linear response properties are also available for spin-restricted, high-spin DFT (*vide infra*).

2. **NEVPT2:** Dalton 2.0 allows for the calculation of second-order MP2 energy corrections to an MCSCF reference wave function. The approach used here, the "n-electron valence space second-order perturbation theory" approach (NEVPT2) [5, 6, 7], is similar to CASPT2 [8], but is based on a two-electron zeroth-order Hamiltonian (the Dyall Hamiltonian [9]) and thus rarely displays problems with intruder states.

3. **R12 methods:** The R12 approach for obtaining basis-set limit MP2 energies are available in a couple of approximations [10, 11].

4. **Excited-state gradients:** Excited-state gradients are available for singlet excited states for a Hartree–Fock reference wave function, calculated as the single residue of

the quadratic response function. The excited-state gradients can be used to optimzie the structure of excited states using first-order optimization schemes.

5. **Absorption in linear response:** The linear reponse code now allows finite lifetimes for excited states [12], allowing both scattering and absorptions processes to be taken into account simultaneously.

6. **Improvements for 64-bit machines:** The code can now also calculate two-electron spin–orbit integrals as well as do AO2MO transformations for more than 255 basis functions also on 64-bit machines.

7. **Extensions to the coupled cluster code:** CCSD(T) energies and analytical gradients are now available.

8. **Orbital exponent gradients:** Gradients of the orbital exponents can now be calculated for closed-shell molecular species [**?**].

9. **Spin–spin coupling constants:** Allows for the calculation of coupling constants to one single nucleus.

10. **Changes to DALTON.INP:** To harmonize the input for single-point and geometry optimization calculations, the labels for calculating molecular properties during a geometry optimization is changed such that

Old label	New label
**START	**START
***PROPERTIES	**EACH STEP
***FINAL	**PROPERTIES

11. **New MOLECULE input format:** There is now a new input structure for the MOLECULE input file. The new format is keyword driven and contains no fixed-format input lines unless the basis set is explicitly given in the input file. However, with the exception of the `ATOMBASIS` option, the input file is fully backward compatible with the old input format. For more information, see Sec. 23.

12. **Single input file:** In addition to the new input format for DALTON, DALTON now also allows the entire input to be given in a single file. This file must be started by the MOLECULE input on the first line, and the DALTON input is placed at the end of the input file. An optimization of methane may thus look like:

```
BASIS
aug-cc-pVDZ
```

```
CH4  molecule. Basis: aug-cc-pVDZ.
Geometry from JCP 112, 393 (2000).
Atomtypes=2 Generators=2 Y X Angstrom
Charge=6.0 Atoms=1
C     0.000000        0.000000        0.000000
Charge=1.0 Atoms=2
H     0.889981273     0.000000000  -0.629311793
H     0.000000000     0.889981273   0.629311793

**DALTON INPUT
.OPTIMIZE
**WAVE FUNCTION
.HF
*END OF INPUT
```

This single input file has to be named `foo.dal`, and only a single filename is given as arguments to the DALTON script, that is

```
> ./dalton foo
```

13. **More basis set directories:** DALTON will now search for basis sets in any user specified directories, the job directory, and in the DALTON basis set library. DALTON 1.2 would search in only one directory, either a directory specified by the user or the DALTON basis set library.

14. **Changes to basis set names:** The following basis sets have been renamed from DALTON 1.2, largely for consistency with the EMSL basis set order form. Note that spaces are not permitted in basis set names.

Old Label	New Label
DunningDZ	DZ(Dunning)
DunningTZ	TZ(Dunning)
SVP(Dunning-Hay)+diffuse	SVP+Diffuse(Dunning-Hay)
AhlrichsVDZ	Ahlrichs-VDZ
AhlrichsVTZ	Ahlrichs-VTZ
japrtano	Almlof-Taylor-ANO
daug-cc-pVXZ (also t,q)	d-aug-cc-pVXZ (t-,q-)
daug-cc-pCVXZ (also t,q)	d-aug-cc-pCVXZ (t-,q-)
sadlej	Sadlej-pVTZ
sad-J	Sadlej-pVTZ-J

15. **Depricated features: PVM no longer supported:** The parallel version no longer supports PVM as a message passing interface. Currently Dalton will only correctly install a parallel version using MPI. Note also that if a parallel DALTON is requested, the executable can both run parallel and sequential calculations.

16. **Numerical Hessian in VROA/Raman calculations:** The analytical Hessian is no longer required for the calculation of Raman intensities and Vibrational Raman optical activities. Instead, the Hessian may be calculated numerically using the analytical gradients. This can be achieved adding the keyword `.NUMHES` in the three input modules `**START`, `**EACH S` and `**PROPER`.

2.2   New features in Dalton 1.2

Dalton 1.2 includes several large modules that have been added in order to increase the possibilities for user applications of the code. In addition, several improvements in the general performance of the code has been made relative to Dalton 1.1. The most important new features and improvements are presented here.

1. **Coupled-cluster theory:** The most important and largest improvement is the addition of a complete integral-direct coupled cluster module, capable of calculating all common coupled cluster models, most of them in an integral-direct manner. Frequency-dependent response functions have also been implemented up to the cubic response functions for the CCS, CC2, and CCSD models. Several models for calculating molecular properties of excited states are also available. Geometry optimization using analytical gradients for electronic ground states using first-order optimization methods is also available.

2. **Non-equilibrium solvation** Non-equilibrium solvation linear, quadratic, and cubic response functions have been implemented, and this makes for instance solvent shifts on excitation energies directly accessible in a one-step calculation. Solvent effects on non-linear optical processes can also be calculated with this approach.

3. **SOPPA** The Second-Order Polarization Propagator Approach (SOPPA) has been introduced in the ABACUS module, allowing for convenient and easy calculations of second-order properties such as polarizabilities, indirect spin–spin coupling constants, nuclear shieldings and magnetizabilities, the latter two only without London orbitals.

4. **CSF** Configurations State Functions can now be used also in property calculations in both the RESPONSE and ABACUS modules, allowing for better control of the spin-state of the reference wave function also in property calculations.

5. **AMFI** An Atomic Mean-Field Approximation of the spin-orbit integrals have been implemented. Even though this mean-field spin-orbit approximation only involves one-electron integrals, it has proven to be a very good approximation to the full one- and two-electron Breit-Pauli spin–orbit operator.

6. **ECP** Effective Core Potentials can now be used for calculations of energies and response functions in which the basis set does not depend on the applied perturbations. That is, molecular gradients and Hessians, as well as London orbital-based magnetic properties, can not be calculated with ECPs.

7. **Douglas-Kroll** The Douglas-Kroll scalar relativistic Hamiltonian has been implemented to second order. It can be used to account for scalar relativistic effects in molecular energies and response functions, and although molecular gradients and Hessians can be calculated, they are not strictly correct, as they use the non-relativistic derivatives of the one-electron Hamiltonian.

8. **Linear coupling model** The linear coupling model can be used for estimating Franck-Condon factors.

9. **Vibrational averaging** A new approach for calculating zero-point [14, 15] or temperature-dependent [16] vibrational averaged molecular properties. For general properties, this have to be in a two-step procedure in which the first step determines the vibrational averaged geometry, and the second calculates the average of the molecular property over a harmonic oscillator approximation for the different vibrational modes.

10. **Magnetic Circular Dichroism** The $\mathcal{B}(0 \rightarrow f)$ term contributing to magnetic circular dichroism (MCD) has been implemented [17].

11. **Two-photon absorption** Although possible to calculate with Dalton 1.1, the input required for calculating two-photon absorption has been significantly improved.

12. **Solvent geometry optimizations** Geometry optimizations using the spherical cavity model can now be done using symmetry. However, only the second-order geometry optimization routines in the `*WALK` module will be able to do geometry optimizations with the solvent model.

13. **File splitting** For file systems where the maximum file length is 2 Gb, DALTON will automatically split large files so that calculations can be performed on these systems even though some files may be longer than 2 Gb. Currently the implementation limits the maximum file size to 22 Gb.

14. **Restart features in** RESPONSE Assuming that the files `RESULTS.RSP` and `RSPVEC` are available, the RESPONSE program can now be restarted at any point of a linear,

quadratic or cubic response calculation. At the most a few micro-iterations of a solution of a linear set of equations will be lost. One can now also reuse the converged response vectors of a quadratic response calculation in a cubic response vector, making it significantly more computationally efficient to determine both linear, quadratic and cubic response functions using SCF or MCSCF wave functions.

Part I

DALTON Installation Guide

Chapter 3

Installation

3.1 Hardware/software supported

DALTON can be run on a variety of systems running the UNIX operating system. The current release of the program supports Cray-UNICOS, Cray-T3D/E, HP-UX, IBM-AIX, IBM-SP2, Linux using g77, pgf77 or ifc, SGI-IRIX, Sun, MacOS X (darwin) using xlf/xlc or g77/gcc, and DEC-Alpha. We furthermore note that response calculations involving the spin–orbit operator only will work on some of these computers (more details are given in the file `dalton/test/KNOWN_PROBLEMS`).

The program is written in FORTRAN 77 and C , with machine dependencies isolated using C preprocessor directives. All floating-point computations are performed in 64-bit precision, but if 32-bit integers are available the code will take advantage of this to reduce storage requirements in some sections.

The program should be easily portable to other UNIX platforms: indeed, the IBM-AIX version may very well work without change on typical 32-bit UNIX workstations. Users who port the codes to other platforms are encouraged to communicate any required changes in the original source with the appropriate C preprocessor directives to the authors.

For installation under MacOS X with the G5 architecture, note that the current release of DALTON requires OSX 10.3, XTools version 1.5 and gcc version 3.3, all available from the Apple website. We recommend the use of the IBM xlf (v8.1) and xlc (v6.0) compilers, as development and testing is with these compilers.

3.2 Source files

DALTON is distributed as a `tar` file obtainable from the DALTON homepage (`http://www.kjemi.uio.no/software/dalton/dalton.html`) if the license agreement for the program has been completed and returned to the authors. If you have accessed this

documentation off the `tar` file you will already know how to extract the required directory structure, but for completeness, assuming the `tar` file is called `dalton.tar.gz`, the commands

```
gunzip dalton.tar.gz
tar xf dalton.tar
```

will produce the following subdirectory structure in the current directory:

```
dalton/abacus      dalton/dft         dalton/include     dalton/test
dalton/amfi        dalton/Doc         dalton/pdpack      dalton/tools
dalton/basis       dalton/eri         dalton/rsp
dalton/cc          dalton/gp          dalton/sirius
```

Most of the subdirectories contain source code for the different sections constituting the program (`abacus`, `amfi`, `cc`, `dft`, `eri`, `gp`, `rsp` and `sirius`). Furthermore, there's a directory containing various public domain routines (`pdpack`), a directory with the necessary include files containing common blocks and machine dependent routines (`include`), a directory containing all the basis sets supplied with this distribution (`basis`), a fairly large set of test jobs including reference output files (`test`), a directory containing some useful pre- and post-processing programs supplied to us from various users (`tools`), and finally this documentation (`Doc`).

In addition to the directories, the main dalton directory will contain several files including a shell script (`configure`) which will build a suitable `Makefile.config` for use when installing the program. The `configure` script will also create a `Makefile` and a run script `bin/dalton` from the skeletal files (`Makefile.in` and `dalton.gnr`) that are present in the directory.

3.3   Installing the program using the Makefile

The program is easily installed through the use of the supplied `configure` script. Based on the architecture supplied by the user, the script will try to build a suitable `Makefile.config` on the basis of what kind of mathematical libraries are found, and user input. Thus, to execute the script, type

```
> ./configure -hosttype
```

where hosttype may be one of

aix	darwin	linux-alpha
cray	dec-alpha	nec
cray-t90	hal	sgi
cray-t3d	hp	sun
cray-t3e	linux	

If no hosttype is specified, the script will try to guess what sort of machine it is running on. This usually works fine for most common platforms.

Although this script in most cases is capable of making a correct `Makefile.config`, we always recommend users to check the created `Makefile.config` against local system set-up. This applies in particular in case of parallel installations, where the user manually *has* to specify certain variables like the path for the MPI-libraries.

During the execution of the `configure` script, you will be asked a few questions, most of which require a quite obvious answer. Let us only comment upon four of the questions asked:

1. Scratch memory size: Dalton uses approximately 6.5 Mwords (54 Mbytes) in static memory allocations. The program defines a large scratch-memory array, from which it allocates space for temporary arrays during the execution of the program. This value is given in Words, and should be chosen according to available memory on your computer. `WRKMEM` may be changed at execution time by supplying a different value for `WRKMEM` through the shell script running DALTON.

2. Default basis set library location: This defines the directory where the program will look for the basis sets supplied with the distribution, and this need to be changed according to the local directory structure. We recommend that the basis sets in this directory are *not* changed, but that changes to the basis set rather is done in a separate directory, and then supply this basis set directory to the program at execution time using the `dalton -b basdir` option.

   NOTE TO SYSTEM ADMINISTRATORS: Supplied with DALTON is an extensive basis set library. This basis set directory must be made readable to all users. END NOTE

3. Default scratch space: Determines the default head scratch directory where temporary files will be placed. This value will be put in the `dalton` run script. However, note that jobs will be run in a subdirectory of this head scratch-directory, according to the name of the job files. If `/work` or `/scratch` is defined in the local directory structure, the script will normally suggest `/work/$USER` or `/scratch/$USER` as default head scratch space.

4. Default install directory: Denotes the directory where the DALTON executable and the `dalton` run script will be moved to. Default: a subdirectory `./bin` to the main DALTON directory.

Compiler options will be supplied in `Makefile.config`, in the manner we use ourselves. *These options often do not include aggressive optimization (for instance on the*

Cray), as it is our experience that the code is optimized incorrectly if the optimization is too aggressive. The proper options to the C preprocessor for an ordinary installation of the program are also supplied with `Makefile.config`.

The `configure` script attempts to detect mathematical libraries available on the system and use them whenever possible. DALTON can use third-party BLAS and LA-PACK libraries or libraries providing equivalent functionality like ESSL on AIX, COM-PLIB.sgimath on SGI, DXML and CXML on Alpha architectures, ATLAS, ACML, MKL on Intel architectures, VECLIB on HP/UX. You will be asked which one you want of the found libraries. You can also point the `configure` script to the location of the preferred libraries by setting `LIBDIR` environment variable before running the script. For example, if one has high-performance BLAS routines in `mylibs` subdirectory of the user's home directory, `configure` script can be instructed to use them as follows:

```
env LIBDIR=\$HOME/mylibs ./configure
```

NOTE: If problems with I/O are experienced on any computer, manifesting themselves as error messages saying that a read statement has passed the EOF mark, the *.F files in the gp directories should be touched, and the code rebuilt using the additional C preprocessor directive `VAR_MFDS`.

When `Makefile.config` has been properly created and checked to agree with local system set-up, all that is needed to build an executable version of the code is to type (in the same directory as the `Makefile.config` file):

```
> make
```

For MacOS X, the program may be installed either with the IBM xlf compiler (version 8.1 at the time of release) for best performance or with the g77 compiler (version 3.4.2 at the time of release). Be sure to use the latest Xcode from `http://developer.apple.com/`. The g77 compiler may be obtained from `http://hpc.sourceforge.net/`. With the g77 compiler, the program may be installed as configured by default (-O2 optimization level). However, for slightly better performance, perform the following two-step process: First, make a copy of the original `Makefile.config` file. Next, in `Makefile.config`, change -O2 to -O3 and type

```
> make -k
```

Finally, revert to the original `Makefile.config` file and then type

```
> make
```

This process results in a code where -O3 has been used wherever possible.

3.4 Running the DALTON test suite

To check that DALTON has been successfully installed, a fairly elaborate automatic test suite is provided in the distribution. A test script, all the test jobs and reference output files can be found in the `dalton/test` directory. It is highly recommended that all these tests be run once the program has been compiled. Depending on your hardware, this usually takes 2—12 hours.

The tests can be run one by one or in groups, by using the test script `TEST`. Try `TEST -h` to see the different options this script takes. Also, have a look at `CONTENTS` for short descriptions of the various tests. To run the complete test suite, simply go to the `dalton/test` directory and type:

```
> ./TEST all
```

You can follow the progress of the tests directly, but all messages are also printed to a log (`TESTLOG` by default). After all the tests have completed you should hopefully be presented with the message "ALL TESTS ENDED PROPERLY!". If not, you will be given a list of the tests that failed to run correctly. Please consult the file `KNOWN_PROBLEMS` too see if these tests have documented problems on your particular platform.

Any tests that fail will leave behind the `.mol` and `.dal` input-files (these are described in more detail in Chapter 5), and the output-file from the test calculation which will have the extension `.log` (DALTON output-files usually have the extension `.out`). For all successful tests these files, as well as some other auxiliary files, will be deleted as soon as the output has been checked, unless `TEST` is being run with the option `-keep`.

If most of the tests fail, it is quite likely that there's something wrong with the installation. Look carefully through `Makefile.config`, and consider turning down or even off optimization.

If there's only a few tests that fail, and DALTON seems to exit normally in each case, there may just be some issues with numerical accuracy. Different machines give slightly different results, and while we've tried to allow for some slack in the tests, it may be that your machine yields numbers just outside the intervals we've specified as acceptable. A closer comparison of the results with numbers in the test script and/or the reference output files should reveal whether this is actually the case. If numerical (in)accuracy is the culprit, feel free to send your output-file(s) to `dalton-admin@kjemi.uio.no` so that we can adjust the numerical intervals accordingly.

Chapter 4

Maintenance

4.1   Memory requirements

DALTON is designed to exit gracefully if there is not enough memory at some stage of the calculation. Wherever possible, it will indicate how much additional memory is required for this particular section. This does not exclude the possibility that the calculation will use even more memory at later stages of the calculation. If DALTON aborts in the response calculation, it may be possible to proceed by resetting the number of perturbations handled simultaneously (see Sec. 25.1.15). Otherwise, some redimensioning will be necessary.

The program use approximately 6.5 Mw of memory allocated as common blocks and similar data structures. Most of the memory used during the calculation is taken from a large work array. The size of this work array is supplied to the program as a shell variable, `MEMWRK`. How this is done is exemplified in the chapter on how to get started (Chapter 5). This makes it easy to give the program the memory needed for a given calculation. If no value to the shell variable is supplied, the program will use the default value which is determined at installation time from the variable `INSTALL_WRKMEM` in the preprocessor directives.

4.1.1   Redimensioning DALTON

Most of the DALTON program has in principle no limitations on the size of the problem one can handle, apart from physical memory requirements and available disc space. The default version of the program can handle 1200 basis functions, 120 nuclei and 400 occupied orbitals and up to i functions in the basis set. However, most such settings are adjustable to suit your own needs, at the expense of larger (or smaller of course) static memory requirements. The following list contains the parameters it is most likely that you will find a need for changing with the corresponding include file in which the variable appears and a short description of what it defines. However, we note that the program sometimes will require

the user to change other parameters not included in this list, and these variables are then easily found in the include/ directory by "grep'ing for" its occurrence. We also point out that whenever a change is made to one or more include files, it is strongly recommended to do `make depend` and then rebuild the program using the command `make`.

MAXOCC	maxorb.h	Maximum number of occupied orbitals.
MXCORB	maxorb.h	Maximum number of contracted atomic orbitals.
MAXWOP	infvar.h	Maximum number of orbital rotations.
MXPST	mxpdim.h	The parameter MXPST is the largest allowed number of string types. A string type is defined by a given number of electrons in RAS1 and RAS3. So for a normal CAS or FCI calculation there is only a single string type (all electrons in for example RAS2). The number of string types become large when one allows a large number of electrons in RAS3 and a large number of holes on RAS1. If the number of string types in a given calculation exceeds MXPST the program stops, and informs the user to raise MXPST.
MXCENT	mxcent.h	Maximum number of nuclear centers + point charges + ghost orbital centers.
MXQN	maxaqn.h	Maximum angular momentum + 1 $(l+1)$ in basis set.
MXAOVC	aovec.h	Maximum number of basis function blocks for a given nuclear center.

4.2 New versions, patches

New versions will be announced on the DALTON homepage (`http://www.kjemi.uio.no/software/dalton/dalton.html` and the DALTON mailing-list (`dalton-users@kjemi.uio.no`).[1]

Inbetween releases, bug fixes will be distributed as patches to an existing version. Although not frequently used in the past, we believe this to be a more common approach in the future. New patches will be freely available from the DALTON homepage, and will be announced on the DALTON mailing list. Patches will normally be distributed in order to correct significant errors. In case of minor changes, explicit instructions on modifying the source file(s) may be given.

In general, users should not make local modifications in the FORTRAN source code, as this usually makes it much harder to incorporate the changes into subsequent versions. It is more convenient to use the C preprocessor code. Indeed, by judicious use of local `#define` variables (this is described in any book on C that also describes the C preprocessor) it is often possible to isolate local modifications completely, making it much easier to apply them to later releases.

---

[1]We no longer support an American mirror site for the DALTON program.

One may be interested in code that is newly developed within the DALTON author group, but not part of any release of the program. In such cases, users are asked to take direct contact with one of the authors of that code, and ask him/her for the new code needed. Such requests are *not* to be sent to `dalton-admin@kjemi.uio.no`. Note also that this may lead to user restrictions of the program not contained in the license agreement signed for a released version of the program.

4.3 Reporting bugs and user support

The DALTON program is distributed to the computational chemistry society with no obligations on the side of the authors. The authors thus take no responsibility for the performance of the code or for the correctness of the results. This distribution policy gives the authors no responsibility for problems experienced by the users when using the DALTON program.

A mailing-list — `dalton-users@kjemi.uio.no` — has been created for the purpose of communication among the authors and the users about new patches and releases, and communication between users of the program. The authors hope that the users will be able to help each others out when problems arise. All authors of the DALTON program also receive mail from the mailing-list, but we do not guarantee that any author will reply to requests posted to the mailing-list. By default, all users signing a license agreement form will be added to this mailing list. If more people use the same licensed version, these users may send mail to `dalton-admin@kjemi.uio.no` asking to be added to this list as well as indicating whose licensed version of the program they are using.

Bug reports are to be reported to `dalton-admin@kjemi.uio.no` and will be dealt with by one of the authors, although no responsibility on the promptness of this response is given. In general, serious bugs that have been reported and fixed will lead to a new patch of the program, distributed from the DALTON homepage.

The DALTON program is developed in an anarchistic fashion, that is, future releases of the DALTON program will contain code related to areas where the authors have had their greatest interest lately. Although user feedback on weak points of the code is welcomed, the likelihood that the authors will comply with these are relatively minor.

We encourage users to supply their own modifications to the extent that such extensions/modifications seem relevant to a larger number of users of the program system. The authors will, however, take the liberty of deciding whether a supplied extension will be incorporated into the official version of the program or not.

Pre- and post-processing programs to the DALTON program are always welcome, and will, if permitted by the author, be distributed along with the DALTON distribution with the proper credits to the author(s) or be made available from the DALTON homepage.

Part II

DALTON User's Guide

Chapter 5

Getting started with DALTON

In this chapter we give an introduction to the two input files needed for doing a calculation with the DALTON program, as well as the shell script file that is supplied with the program for moving relevant files to the scratch-directory and back to the home directory after a calculation has completed. A couple of examples of input files are also provided. Finally, the different output files generated by the program are discussed.

5.1 The DALTON.INP file

The `DALTON.INP` file contains the keywords telling the program what kind of atomic integrals are wanted (HERMIT, in the `**INTEGRALS` input module), what kind of wave function is to be used (SIRIUS, in the `**WAVE FUNCTIONS` input module), what kind of molecular properties are to be evaluated (ABACUS, in the `**PROPERTIES` input module), and finally which response functions that are to be evaluated (RESPONSE, in the `**RESPONSE` input module).

This input file is described in detail in several of the subsequent chapters, and we will only give a few examples of input files here and shortly explain what they do, in order to give the user a flavor of the input and the computational possibilities of DALTON.

5.1.1 A CASSCF geometry optimization

We start by a simple geometry optimization of a water molecule using a Complete Active Space (CAS) wave function, where we use C_{2v} symmetry keeping the 1s orbital on oxygen inactive and distributing the valence electrons into 8 orbitals. At the starting geometry we evaluate the nuclear magnetic shielding constants and the magnetizability of the water molecule, and at the optimized geometry we perform a vibrational analysis and calculate the IR intensities (related to the dipole gradient). The input file for such a calculation will look like:

```
**DALTON INPUT              Must start all input files
.OPTIMIZE                   Request geometry optimization
**WAVE FUNCTIONS            Wave function input
.HF                         We start with HF
.MP2                        Then MP2 (starting orb.  for MCSCF)
.MCSCF                      Request an MCSCF
*SCF INPUT                  HF input
.DOUBLY OCCUPIED
3 1 1 0
*CONFIGURATION INPUT        Input of active space
.SYMMETRY                   Wave function symmetry
1
.SPIN MULTIPLICITY
1                           Singlet
.INACTIVE                   Doubly occupied orbitals
1 0 0 0                     1s on oxygen
.CAS SPACE
4 2 2 0
.ELECTRONS                  Number of electrons to correlate
8                           The valence electrons
**START                     Input for start geometry
.SHIELD                     Nuclear shieldings
.MAGNET                     Magnetizability
**PROPERTIES                Input for optimized geometry
.DIPGRA                     Dipole gradient
.VIBANA                     Vibrational analysis
**END OF DALTON INPUT
```

5.1.2   A RASSCF calculation of NMR parameters

The next input file gives an example of the input needed for evaluating the two parameters determining an NMR spectrum, the nuclear shielding constants and the spin–spin coupling constants. We do this at a fixed geometry, and use a Restricted Active Space (RAS) wave function. In this way we include near-degeneracy effects by inclusion of the appropriate orbitals in the RAS2 space, and also some of the dynamical correlation by single and double excitations from the RAS2 space (and generally also form the RAS1 space) into the RAS3 space.

In this input we once more use water as input molecule, where we keep the 1s orbital inactive, correlate 8 electrons in 6 orbitals (Full-Valence CAS) and in addition allow single- and double-excitations into 7 additional orbitals.

```
**DALTON INPUT
.RUN PROPERTIES              Run integrals, wave function and properties
**WAVE FUNCTIONS
.HF
.MP2
.MCSCF
*SCF INPUT
.DOUBLY OCCUPIED
3 1 1 0
*CONFIGURATION INPUT
.SYMMETRY
1
.SPIN MUL
1
.INACTIVE
1 0 0 0
.RAS1 SPACE                  # orbitals in RAS1 space
0 0 0 0
.RAS2 SPACE                  The ''Cas Space''
3 2 1 0
.RAS3 SPACE
3 1 2 1
.ELECTRONS                   Number of electrons to correlate
8                            The valence electrons
.RAS1 HOLES                  # elect.  that may be removed from RAS1
0 0
.RAS3 ELECTRONS              From 0 to 2 electrons excited into here
0 2
**PROPERTIES                 Input for single geometry run
.SHIELD
.SPIN-S
**END OF DALTON INPUT
```

5.1.3 A parallel cubic response calculation

In this example of a `DALTON.INP` input file, we assume that the program has been compiled with MPI options, and request a SCF calculation of the second hyperpolarizability (γ_{xxxx}) using the cubic response solver of the RESPONSE program. No two-electron integrals will be stored on disc in any part of the calculation. The number of nodes used is determined when submitting the job, and how this is done depends on the computer on which the job is run.

All treatment of symmetry, as well as the Hartree–Fock occupation, is automatically

taken care of by the program, and we thus only need to specify that we are to do a Hartree–Fock calculation.

```
**DALTON INPUT
.RUN RESPONS                Run integrals, wave func. and response
.PARALLEL                   This is a parallel run
**WAVE FUNCTIONS
.HF
**RESPONSE                  Input for the response program
*CUBIC                      Input for cubic response calculation
.DIPLNX                     Only consider x-comp.  of dipole moment
.BFREQ                      Two frequencies for B operator
1
0.0 0.0592                  Frequencies in atomic units
.CFREQ                      One frequency for C operator
1
0.0
.DFREQ                      One frequency for D operator
1
0.0
.THCLR                      Threshold for linear response equations
1.0D-5
**END OF DALTON INPUT
```

5.2   General structure of the DALTON.INP file

The input is divided into modules, which in turn are further divided into submodules. The modules contain the input to the different sections that constitute the DALTON program package. There exists four such sections corresponding to respectively the integral evaluation, the choice of wave function, the choice of static properties and finally the choice of dynamic properties. For each of these sections there exists a module with a name started by two stars, that is:

```
**INTEGRALS
```

```
**WAVE FUNCTIONS
```

```
**PROPERTIES
```

```
**RESPONSE
```

In addition there is a module controlling the DALTON program, which has the input card **DALTON. All input modules will be described in detail in separate chapters in the Reference Manual.

For geometry optimizations, different options may often be wanted for different stages of the geometry optimization, and there exists three different ABACUS input modules for geometry optimizations: `**START`, `**EACH STEP`, `**PROPERTIES`. `**START` gives input for the initial point, `**EACH STEP` gives input options during the optimization of the molecular geometry (and for the initial point if no `**START` was specified and for the final point if no `**PROPERTIES` was given), and `**PROPERTIES` gives input options for the analysis that is to be done at the optimized geometry. In this way one may start a calculation at an experimental geometry, evaluating a set of molecular properties of interest, do an ordinary geometry optimization without any properties evaluated during the geometry optimization, and then recalculate the same set of molecular properties (or others) at the optimized geometry in one single run. Note that if a `**RESPONSE` module present in a geometry optimization, the RESPONSE calculations will only be performed at the optimized geometry.

The input file must end with the keyword `*END OF`.

In a given module, several submodules may be specified, which determine the performance in various parts of the calculation. The different input cards available in the different submodules are started with a dot, *i.e.* an input option should be typed as `.IMAGE`. For most calculations the default settings may be used. The authors have, however, always opted for the largest degree of freedom, giving the user good opportunities for tailoring an input to suit the needs of a given calculation.

Even though we at times write large, self-explanatory keywords in this manual, only the first seven characters (including the `.` or `*`) are significant and will be treated by the program. At any place in the input, comments may be added using `#` or `!` in the first position of the command line. The input is case sensitive and only upper-case characters will be recognized.

A general input structure for DALTON will look like

```
**DALTON INPUT
...
**INTEGRALS
...
**WAVE FUNCTIONS
...
*CONFIGURATION INPUT
...
**START
...
**EACH STEP
...
**PROPERTIES
```

```
...
**RESPONSE
*CUBIC
...
**END OF DALTON INPUT
```

Several submodules can of course be added in each of the above modules. We restate that if RESPONSE is requested in a geometry optimization, the evaluation of the response functions will only be done at the optimized geometry.

Examples of input files can be found in the various chapters describing the calculation of different molecular properties, and these chapters will probably be sufficient for most calculations done with DALTON. However, all possible keywords are described in detail in separate chapters in the Reference Manual. The tests in the `dalton/test` directory also serves to demonstrate how to run different kinds of calculations using DALTON (see the `CONTENTS` file in the test directory).

5.3   The MOLECULE input file

We will not go into great detail here of the input format of the MOLECULE input file, as it is treated thoroughly in a separate chapter, Chapter 23. Here we only present two inputs for water; one using Cartesian coordinate input and automatic symmetry detection and the other using Cartesian coordinates and where we explicitly give the symmetry elements. Finally we show the MOLECULE input for H_2O^{2+} using Z-matrix input. In all cases the basis set library of DALTON has been used, as we assume that most users will exploit the basis sets provided in the basis set library as opposed to punching them in. The punching format for basis sets are described in detail in Sec. 23.2.

Let us first start with an input for water using Cartesian coordinates in bohr, and the very popular 6-31G** basis set.

```
BASIS
6-31G**
Water
using the 6-31G** basis
Atomtypes=2
Charge=8.0 Atoms=1
O     .0000000000        -.2249058930        .0000000000
Charge=1.0 Atoms=2
H    1.4523499293         .8996235720        .0000000000
H   -1.4523499293         .8996235720        .0000000000
```

On the fifth line the number of different atomtypes or more correctly, the number of blocks of atoms, in this case two, oxygen and hydrogen, are given. There are one oxygen with charge 8, and two hydrogens with charge 1. The symmetry of the system will be detected by the program during the input processing unless turned off as shown in the last of the input examples for the MOLECULE input files.

We may also add the symmetry elements of the symmetry group ourselves. We must then remove any symmetry-dependent centers. For the above input this will result in, if we use the yz- (the x axis changes sign during the symmetry operation) and xy- (the z axis changes sign) -planes as symmetry elements:

```
ATOMBASIS
Water
using ANOs specified for each atomtype
Atomtypes=2 Generators=2 X Y
Charge=8.0 Atoms=1 Basis=ano-1 5 4 3 1
O      .0000000000      -.2249058930      .0000000000
Charge=1.0 Atoms=1 Basis=ano-1 4 3 2
H    1.4523499293       .8996235720       .0000000000
```

In the above calculation we used the ANO set of Widmark and coworkers [18, 19]. We use the contraction [5s4p3d1f/4s3p2d] for the oxygen and the hydrogens respectively. We also note the keyword ATOMBASIS which allows different basis functions to be used on different atoms.

Let us now proceed to an input for H_2O^{2+} where we use Z-matrix input. The input will then look like:

```
BASIS
Sadlej-pVTZ
Water - geometry optimization with Sadlej's basis set
Project - Raman
Atomtypes=3  Charge=2 Nosymmetry
ZMAT
O   1 8.0
H   2 1 0.97 1.0
H   3 1 0.97 2 104.5 1.0
```

In addition to the number of atoms in the input (3), we have given the total charge of the molecule (+2), and the keyword Nosymmetry indicates that symmetry is not to be used in this calculation. Note that unlike Cartesian coordinate input, which by default is in atomic units, the Z-matrix input is always to be in Ångström.

5.4 The first calculation with DALTON

If we have made two input files, one corresponding to `DALTON.INP` and one corresponding to the MOLECULE input file, we are ready to do our first calculation. Examples of input files can also be found in the `dalton/test` directory hidden inside the test jobs. You can execute one of the individual test jobs without the `TEST` script, for example:

```
> ./energy_nosymm
```

This will create four files, but right now we are just interested in the `.dal` and the `.mol` files. In this particular example they will be `energy_nosymm.dal` and `energy_nosymm.mol`. Have a look at some of these input files to get a head-on start on running different kinds of DALTON jobs.

Calculations with DALTON is most conveniently done using the supplied shell script `dalton`. Thus, to run a calculation of β (first hyperpolarizability) with input available as `beta.dal` on the HCl molecule, and with molecule input available as `hcl.mol`, you would type

```
> dalton beta hcl
```

assuming `dalton` is available in your path. When the job is finished, the output is copied back as `beta_hcl.out`. In case that the dalton- and molecule-input has the same name we may write `dalton energy_nosymm`, and the corresponding output file will be named `energy_nosymm.out`. In addition, the program will copy back a file named `beta_hcl.tar.gz`. This file contains, in tar'ed and gzip'ed form, a number of useful interface and post-processing files needed for post-DALTON programs, or needed when restarting calculations.

There are several options to this script, which can be viewed by typing `dalton -h` or just `dalton`. These options include:

-b directory Prepend this directory to the list of directories where the program should look for basis sets. Needed in case you want to use local modifications of a given basis. The job directory and the DALTON basis set library will always be included in the basis set directory list (in that search order).

  -d Removes the contents of the scratch directory before a calculation starts in order to avoid inconsistencies between files.

  -D Do not remove the content of the scratch directory. By default the scratch directory will be deleted. However, in order to do a restart you may want to keep all files in this directory, and you then need to add the -D option when submitting the job.

-e ext Change the extension to the output file from `.out` to `.ext`.

-f Copy the gzipped-file containing a variety of useful interface files from your home directory to the scratch directory before a calculation starts. This is needed in order to be able to restart DALTON calculations, or if you want to use converged response vectors in a different response calculations for the same molecule.

-M mb Change the default size of the scratch memory space to `mb` Mbytes.

-N nodes Number of nodes to be used in a parallel MPI calculation controlled by running mpirun (or using mpiexec).

-lam file Call "lamboot file" and "wipe file" before and after mpirun.

-o filename Redirect the output normally printed in the `DALTON.OUT` file in the temporary directory to the file "filename" in the WRKDIR directory. On a computer system with distributed scratch disk but a commonly mounted home directory, this allows you to follow the calculation without having to log into the compute nodes.

-t tmpdir Change the `TMPDIR` (the scratch disc area) to `tmpdir` from the default scratch directory determined at install-time.

-w wrkdir Change the working directory to "wrkdir", that is, change the directory in which the program searches for input files and places the `DALTON.OUT` file to "wrkdir".

In most cases, the `DALTON.OUT` file will contain all the information needed about a given calculations. However, in certain cases, additional information may be wanted, and this is contained in various sets of auxiliary files. These files are copied back in the tar'ed and gzip'ed file. This file may include the following set of different files:

`DALTON.BAS` Contains a dump of a complete molecule input file. This file take maximum advantage of formatted input, yet differences may occur compared to the basis sets obtained from the basis set library due to the restricted number of digits available in the standard-format output.

`DALTON.CM` An output file that contains the most essential information needed for calculation of shielding polarizabilities and magnetizability polarizabilities. Most easily used together with the analyzing program `ODCPRG.f` supplied in the tools directory.

`DALTON.HES` If the keyword `.HESPUN` has been specified in the `*VIBANA` input module, the molecular Hessian will be written to this file in a standard format, which may be used as a start Hessian in an first-order geometry optimization, or as input to a ROA or VCD analysis with different basis sets/level of correlation for the intensity operators and the force field. See also the `FChk2HES.f` program in the tools directory.

`DALTON.IRC` Contains information obtained from an Intrinsic Reaction Coordinate (IRC) calculation, as described in Sec. 7.2.1.

`DALTON.MOL` Contains the information needed by the `MOLPLT`-program for visualizing the molecular geometry. The `MOLPLT`-program is distributed with the GAMESS-US program package.

`DALTON.MOPUN` Contains the molecular-orbital coefficients printed in a standard format allowing the transfer of molecular orbitals coefficients from one computer to another.

`DALTON.NCA` Contains the information needed by the `MOLPLT`-program for visualizing normal coordinates.

`DALTON.ORB` Contains information about basis set and MO-coefficients so that MO density plots may be generated using the `PLTORB` program that comes with the GAMESS-US distribution. Currently not supported.

`DALTON.TRJ` Contains trajectory information from a direct dynamics calculation as described in Sec. 7.2.2.

`DALTON.WLK` Contains information from the walk-procedure, and is needed when restarting a walk (e.g. a numerical differentiation).

`molden.inp` Contains the input required for visualizing the results of the calculation using the MOLDEN program (http://http://www.cmbi.ru.nl/molden/molden.html). Please also note that Jmol (http://jmol.sourceforge.net) also can visualize many aspects of a DALTON calculation, including an intrisic reaction coordinate calculation. Jmol uses the DALTON output file (DALTON.OUT) for it's visualization.

`RESULTS.RSP` Contains a brief summary of the results obtained form the response functions that have finished. The program may use this information to skip response equations that have already been solved (for instance if the calculation crashed for some reason during the calculation of a set of cubic response functions).

`RSPVEC` Contains the converged response equations. The program may use this to avoid repeating linear response equations. Thus, one may use the converged response vectors of a linear response equation in the calculation of quadratic response function, and there may then be no need to solve additional response equations.

`SIRIUS.IFC` Interface file between the wave function part of the program and the property modules. Contains all the information required about the optimized molecular wave function.

SIRIUS.RST Contains restart information needed in case one needs to restart the wave function part of the program.

Chapter 6

Getting the wave function you want

Currently the following wave function types are implemented in DALTON:

RHF Closed-shell singlet and spin-restricted high-spin Hartree–Fock (SCF, self-consistent field).

DFT Density functional theory, closed shell and spin-restricted high-spin Kohn–Sham DFT, allowing for both local density functionals (LDA), generalized gradient approximation (GGA) functionals and hybrid functionals..

MP2 Møller-Plesset second-order perturbation theory which is based on closed-shell restricted Hartree–Fock. Energy, second-order one-electron density matrix and first-order wave function (for SOPPA) may be calculated, as well as first-order geometry optimizations using numerical gradients. MP2 is also part of CC (see Chapter 27). First-order properties and geometry optimization are available through this module.

CI Configuration interaction. Two types of configuration selection schemes are implemented: the complete active space model (CAS) and the restricted active space model (RAS).

CC Coupled Cluster. For information on how to run this kind of calculation, we refer to Chapters 20 and 27.

MCSCF Multiconfiguration self-consistent field based on the complete active space model (CASSCF) and the restricted active space model (RASSCF).

SIRIUS is the part of the code that computes all of these wave functions, with the exception of the coupled cluster wave functions. This part of the DALTON manual discusses aspects of converging the wave function you want.

6.1 Necessary input to SIRIUS

In order for DALTON to perform a calculation for a given wave function a minimum amount of information is required. This information is collected in the `**WAVE FUNCTIONS` input module. Fig. 6.1 collects the most essential information.

6.2 An input example for SIRIUS

The number of symmetries and the number of basis orbitals are read from the one-electron integral file. The number of symmetries may also be specified in the `**WAVE FUNCTIONS` input module as a way of obtaining a consistency check between the wave function input section and available integral files.

The following sample input specifies a CASSCF calculation on water with maximum use of defaults:

```
**WAVE FUNCTIONS
.MCSCF               -- specifies that this is an MCSCF calculation
*CONFIGURATION INPUT
.SPIN MULTIPLICITY   -- converge to singlet state
    1
.SYMMETRY            -- converge to state of symmetry 1 (i.e. A1)
    1
.INACTIVE ORBITALS   -- number of doubly occupied orbitals each symmetry
    1   0   0   0
.CAS SPACE           -- selects CAS and defines the active orbitals.
    4   2   2   0
.ELECTRONS           -- number of electrons distributed in the active
    8                   orbitals.
*ORBITAL INPUT
.MOSTART             -- initial orbitals are orbitals from a previous run
 NEWORB                 (for example MP2 orbitals)
**(name of next input module)
```

Comparing to the list of necessary input information in Fig. 6.1, the following points may be noted: The number of basis functions are read from the one-electron integral file. Because no orbitals are going to be deleted, the number of molecular orbitals will be the same as the number of basis functions. No state has been specified as ground state calculation is the default. The default convergence threshold of 1.0D-5 is used.

```
   Calculation type(s): RHF, MP2, CI, MCSCF, CC.

Number of symmetries (needed only for MCSCF wave functions)
If MCSCF or CI calculation then
   Symmetry of wave function.
   Spin multiplicity of wave function.
else if one open shell RHF
   Symmetry of open shell, doublet spin multiplicity
else
   Totally symmetric, singlet spin multiplicity
end if

Number of basis orbitals.
Number of molecular orbitals.

Number of inactive orbitals in each symmetry.
If high-spin open shell RHF then
   Number of singly occupied orbitals in each symmetry
else if MCSCF or CI calculation then
   Number of active electrons.
   If CAS calculation desired then
      Number of active orbitals in each symmetry.
   else if RAS calculation desired then
      Number of active orbitals in RAS1 space in each symmetry.
      Number of active orbitals in RAS2 space in each symmetry.
      Number of active orbitals in RAS3 space in each symmetry.
      Limits on number of electrons in RAS1 and RAS3.
   end if
end if

if not MP2 then convergence threshold.

how to begin the calculation (initial guess).
```

Figure 6.1: Necessary input information for determining the wave function.

6.3   Hints on the structure of the input for the `**WAVE FUNCTIONS` input module

The following list attempts to show the interdependence of the various input modules through indentation (*R*: required input) :

```
**WAVE FUNCTIONS
  .TITLE
  .HF
    *SCF INPUT
      .DOUBLY OCCUPIED
      ...
    *ORBITAL INPUT        (2)
    *STEP CONTROL         (if QCSCF)
  .DFT
    Functional name
    *SCF INPUT
      .DOUBLY OCCUPIED
    *DFT INPUT
      ...
    *ORBITAL INPUT        (2)
    *STEP CONTROL         (if QCSCF)
  .MP2
    *MP2 INPUT
    *ORBITAL INPUT        (2)
    *SCF INPUT
      .DOUBLY OCCUPIED
  .CI
    *CONFIGURATION INPUT  (1)
    *ORBITAL INPUT        (2)
    *CI VECTOR            (3)
    *CI INPUT
  .MCSCF
    *CONFIGURATION INPUT  (1)
    *ORBITAL INPUT        (2)
    *CI VECTOR            (3)
    *OPTIMIZATION
    *STEP CONTROL
  .CC
```

```
      *CC INPUT
   .STOP
   .RESTART
   .INTERFACE
   .PRINT
*HAMILTONIAN
*TRANSFORMATION
*POPULATION ANALYSIS
*PRINT LEVELS
*AUXILIARY INPUT
```

Notes:

```
(1) *CONFIGURATION INPUT
      .SPIN MULTIPLICITY (*R*)
      .SYMMETRY (*R*)
      .INACTIVE ORBITALS (*R*)

      .ELECTRONS (*R*)
      Either
        .CAS SPACE
      or some or all of
        .RAS1 SPACE
        .RAS2 SPACE
        .RAS3 SPACE
        .RAS1 HOLES or .RAS1 ELECTRONS
        .RAS3 ELECTRONS
      are required.

(2) *ORBITAL INPUT
      .MOSTART   ! default Huckel guess if available, otherwise H1DIAG

      .SYMMETRIC ORTHONORMALIZATION
      .GRAM-SCHMIDT ORTHONORMALIZATION

      .FROZEN CORE ORBITALS
      .FREEZE ORBITALS
```

```
        .5D7F9G
        .AO DELETE
        .DELETE

        .REORDER

        .PUNCHINPUTORBITALS
        .PUNCHOUTPUTORBITALS

(3) *CI VECTOR
        One and only one of the following three
          .STARTHDIAGONAL  ! default
          .STARTOLDCI
          .SELECT CONFIGURATION

        .CNO START
```

6.4 How to restart a wave function calculation

It is possible to restart after the last finished macro iteration in (MC)SCF, provided the `SIRIUS.RST` file was saved. You will then at most loose part of one macro iteration in case of system crashes, disks running full, or other irregularities. In general the only change needed to the wave function input is to add the `.RESTART` keyword under `**WAVE FUNCTIONS`. However, if the original job was a multistep calculation involving *e.g.* SCF, MP2 and MCSCF wave functions, restarting the MCSCF part requires the removal of the "HF" and "MP2" keywords in the input file. If the correct two-electron integrals over molecular orbitals are available, you can skip the integral transformation in the beginning of the macro iteration by means of the `.OLD TRANSFORMATION` keyword under `*TRANSFORMATION`. When resubmitting the job, you must make sure that the correct `SIRIUS.RST` file is made available to the job.

You can also use the restart feature if you find it desirable to converge the wave function sharper than specified in the original input. By default, SCF wave functions are converged to 1.0D-6 in the gradient norm, whereas MCSCF and CI wave functions are converged to 1.0D-5 in the gradient norm.

6.5   Transfer of molecular orbitals between different computers

In order to be able to transfer molecular orbital coefficients between different computer systems, an option is provided for formatted punching and formatted reading of the molecular orbital coefficients. The options are

<p align="center">.PUNCHOUTPUTORBITALS or .PUNCHINPUTORBITALS</p>

in input group *ORBITAL INPUT. The first option is used to punch the orbitals at the end of the optimization, while the second option is used to punch some molecular orbitals already available on the SIRIUS.RST file, for instance the converged orbitals from a previous calculation. The orbitals can then be transferred to another computer and appended to the SIRIUS input there. The orbitals may then be read by DALTON using the .MOSTART followed by .FORM18 option on the next line (with this option DALTON will, after having finished the **WAVE FUNCTIONS input module, search the input file for either the **MOLORB keyword or the **NATORB keyword and expect the orbital coefficients to follow immediately afterwards).

6.6   Wave function input examples

Those who are used to the old SIRIUS input will not experience any dramatic changes with respect to the DALTON input, although a few minor differences exist in order to create a more user-friendly environment. The input sections are the same as before starting with a wavefunction specification, currently, including options for SCF, DFT, MP2, MCSCF, and CC reference states. Depending on this choice of reference state the following input sections take different form, from the simplest SCF input to the more complex MCSCF input. Minimal SCF and SCF+MP2 using the new feature of generating the HF occupation on the basis of an initial Hückel calculation and then possibly change the occupation during the first DIIS iterations will be shown. There will also be an example showing how an old SIRIUS.RST file is used for restart in the input. In this case the Hartree–Fock occupation will be read from the file SIRIUS.RST and used as initial Hartree–Fock occupation. The HF occupation is needed for an MCSCF calculation, as this is anyway determined when establishing a suitably chosen active space. An example of an MCSCF-CAS input without starting orbitals will be given, as well as an MCSCF-RAS with starting orbitals. We note that an MCSCF wave function by default will be optimized using spin-adapted configurations (CSFs). The three following examples illustrate calculations on and excited state, on a core-hole state with frozen core orbital and on a core-hole state with relaxed core orbital. The last example shows an MCSCF calculation of non-equilibrium solvation energy. For examples on how to run CC wave functions, we refer to Chapter 20.

The following input example gives a minimal SCF input with default starting orbitals (that is, Hückel guess), and automatic Hartree–Fock occupation, first based on the Hückel guess, and then updated during the DIIS iterations:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

The following input give a minimal input for an MP2 calculation using all default settings for the Hartree–Fock calculation (see previous example):

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.HF
.MP2
**END OF DALTON INPUT
```

If we would like to calculate molecular properties in several geometries, we may take advantage of the fact the molecular orbitals at the previous geometry probably is quite close to the optimized MOs at the new geometry, and thus restart from the MOs contained in the SIRIUS.RST file, as indicated in the following example:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.HF
*ORBITAL INPUT
.MOSTART
 NEWORB
**END OF DALTON INPUT
```

Running DFT calculations is very similar to Hartree-Fock, except for the specification of the functional: The .DFT keyword is followed by a line describing the functional to be used. The simple input below will start a B3LYP geometry optimization:

```
**DALTON INPUT
.OPTIMIZE
```

```
**WAVE FUNCTIONS
.DFT
 B3LYP
**END OF DALTON INPUT
```

We finish this list of examples with a high-spin open-shell DFT input. We need to specify the number of singly as well as doubly occupied orbitals in each symmetry in the `**SCF INPUT` section using the `.SINGLY OCCUPIED` and `.DOUBLY OCCUPIED` keywords respectively (here assuming D_{2h} symmetry):

```
**DALTON
.RUN WAVE FUNCTIONS
**WAVE FUNCTION
.DFT
 B3LYP
*SCF INPUT
.DOUBLY OCCUPIED
 3 1 1 0 2 0 0 0
.SINGLY OCCUPIED
 0 0 0 0 0 1 1 0
**END OF DALTON INPUT
```

A complete list of available functionals can be found in the Reference Guide, see Sec. 24.2.7

Reference literature:

Restricted-step second-order MCSCF: H.J.Aa.Jensen, and H.Ågren. *Chem.Phys.Lett.* **110**, 140, (1984).

Restricted-step second-order MCSCF: H.J.Aa.Jensen, and H.Ågren. *Chem.Phys.* **104**, 229, (1986).

MP2 natural orbital occupation analysis: H.J.Aa.Jensen, P.Jørgensen, H.Ågren, and J.Olsen. *J.Chem.Phys.* **88**, 3834, (1988); **89**, 5354, (1988).

The next input example gives the necessary input for a Complete Active Space SCF (CASSCF) calculation where we use MP2 to provide starting orbitals for our MCSCF. The active space may for instance be chosen on the basis of an MP2 natural orbital occupation analysis as described in Ref. [20]. The input would be:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
```

```
.HF
.MP2
.MCSCF
*SCF INPUT
.DOUBLY OCCUPIED
 2 0 0 0
*CONFIGURATION INPUT
.SYMMETRY
 1
.SPIN MULTIPLICITY
 1
.INACTIVE ORBITALS
 0 0 0 0
.ELECTRONS (active)
 4
.CAS SPACE
 6 4 4 0
**END OF DALTON INPUT
```

As for Hartree–Fock calculation, we may want to use available molecular orbitals on the SIRIUS.RST file from previous calculations as starting orbitals for our MCSCF as indicated in this input example:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.MCSCF
*ORBITAL INPUT
.MOSTART
 NEWORB
*CONFIGURATION INPUT
.SYMMETRY
 1
.SPIN MULTIPLICITY
 1
.INACTIVE ORBITALS
 0 0 0 0
.ELECTRONS (active)
 4
```

```
.RAS1 SPACE
 2 1 1 0
.RAS2 SPACE
 2 2 2 0
.RAS3 SPACE
 6 4 4 2
.RAS1 ELECTRONS
 0 2
.RAS3 ELECTRONS
 0 2
*OPTIMIZATION
.TRACI
.FOCKONLY
**END OF DALTON INPUT
```

> **Reference literature:**
>
> Optimal orbital trial vectors: H.J.Aa.Jensen, P.Jørgensen, and H.Ågren. *J.Chem.Phys.*, **87**, 451, (1987).
>
> Excited state geometry optimizations: A.Cesar, H.Ågren, T.Helgaker, P.Jørgensen, and H.J.Aa.Jensen. *J.Chem.Phys.*, **95**, 5906, (1991).

The next input describes the optimization of the first excited state of the same symmetry as the ground state. To speed up convergence, we employ optimal orbital trial vectors as described in Ref. [21]. Such an input would look like:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.TITLE
4 2 2 0 CAS on first excited 1A_1 state, converging to 1.D-07
.MCSCF
.NSYM
  4
*CONFIGURATION INPUT
.SYMMETRY
  1                   | same symmetry as ground state
.SPIN MULTIPLICITY
  1
.INACTIVE ORBITALS
```

```
 1 0 0 0
.ELECTRONS (active)
 8
.CAS SPACE
 4 2 2 0
*ORBITAL INPUT
.MOSTART
 NEWORB
*CI VECTOR
.STARTHDIAGONAL     | compute start vector from Hessian CI-diagonal
*OPTIMIZATION
.THRESH
 1.D-07
.SIMULTANEOUS ROOTS
  2 2
.STATE
  2                 | 2 since the first exited state has the same symmetry
!                     as the ground state
.OPTIMAL ORBITAL TRIAL VECTORS
*PRINT LEVELS
.PRINTUNITS
 6 6
.PRINTLEVELS
 5 5
**END OF DALTON INPUT
```

Reference literature:

Core hole: H.Ågren, and H.J.Aa.Jensen. *Chem.Phys.Lett.*, **137**, 431, (1987).

Core hole: H.Ågren, and H.J.Aa.Jensen. *Chem.Phys.*, **172**, 45, (1993).

The next input describes the calculation of a core-hole state of the carbon 1s orbital in carbon monoxide, the first example employing a frozen core:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.TITLE
```

```
C1s core hole state of CO, 4 2 2 0 valence CAS + C1s.
Frozen core orbital calculation.
.MCSCF
.NSYM
  4
*CONFIGURATION INPUT
.SYMMETRY
  1
.SPIN MULTIPLICITY
  2                   | Doublet spin symmetry because of opened core orbital
.INACTIVE ORBITALS
 2 0 0 0              | O1s and O2s orbitals are inactive while the
!                       opened core orbital, C1s, must be active
.ELECTRONS (active)
 9                    | All valence electrons plus the core hole electron
!                       are active
.RAS1 SPACE
 1 0 0 0             | The opened core orbital (NOTE: always only this orb.)
.RAS1 ELECTRONS
 1 1                 | We impose single occupancy for the opened core orbital
.RAS2 SPACE
 4 2 2 0             | Same as the CAS space in the ground state calculation
*OPTIMIZATION
.COREHOLE
 1 2                 | Symmetry of the core orbital and the orbital in this
!                      symmetry with the core hole according to list of input
!                      orbitals. The same thing could be obtained by
!                      reordering the core orbital to the first active orb.
!                      (by .REORDER), and specifying .FREEZE and .NEO ALWAYS
.TRACI
*ORBITAL INPUT
.MOSTART
 NEWORB              | Start from corresponding MCSCF ground state
*CI VECTOR
.STARTHDIAGONAL
**END OF DALTON INPUT
```

whereas we in a calculation where we would allow the core to relax only would require the following changes compared to the previous input, assuming that we start out

from orbitals and CI vectors generated by the previous calculation `.STARTHDIAGONAL` is therefore replaced by `.STARTOLDCI`, and the core orbital has number 3 (instead of 1) in the list of orbitals in the first symmetry. ".CORERELAX" is specified for relaxation of the core orbital using the NR algorithm.

```
*OPTIMIZATION
.COREHOLE
 1 3
.CORERELAX
*CI VEC
.STARTOLDCI
```

Reference literature:

General reference: K.V.Mikkelsen, E.Dalgaard, P.Svanstrøm. *J.Phys.Chem*, **91**, 3081, (1987).

General reference: K.V.Mikkelsen, H.Ågren, H.J.Aa.Jensen, and T.Helgaker. *J.Chem.Phys.*, **89**, 3086, (1988).

Non-equilibrium solvation: K.V.Mikkelsen, A.Cesar, H.Ågren, H.J.Aa.Jensen. *J.Chem.Phys.*, **103**, 9010, (1995).

This example describes calculations for non-equilibrium solvation, where the molecule will be enclosed in a spherical cavity. Usually one starts with a calculation of a reference state (most often the ground state) with equilibrium solvation, using keyword `.INERSFINAL`. The interface file is then used (without user interference) for a non-equilibrium excited state calculation; keyword `.INERSINITIAL`.

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.TITLE
 2-RAS(2p2p') : on F+ (1^D) in Glycol
.MCSCF
.NSYM
 8
*CONFIGURATION INPUT
.SPIN MULTIPLICITY
 1
.SYMMETRY
 1
```

```
.INACTIVE ORBITALS
 1  0  0  0  0  0  0  0
.ELECTRONS (active)
 6
.RAS1 SPACE
 0  0  0  0  0  0  0  0
.RAS2 SPACE
 1  2  2  0  2  0  0  0
.RAS3 SPACE
 8  4  4  3  4  3  3  1
.RAS1 ELECTRONS
 0  0
.RAS3 ELECTRONS
 0  2
*OPTIMIZATION
.NEO ALWAYS
.OPTIMAL ORBITAL TRIAL VECTORS
*ORBITAL INPUT
.MOSTART
 NEWORB
*CI VECTOR
.STARTOLDCI
*SOLVENT
.CAVITY
 2.5133D0
.INERSINITIAL     | initial state inertial polarization
 37.7D0  2.050D0  | static and optic dielectric constants for Glycol
.MAX L
 10
.PRINT
 6
**END OF DALTON INPUT
```

Chapter 7

Potential energy surfaces

This section describes one of the most important features of any quantum chemical software package; locating equilibrium geometries and transition structures of molecules. In DALTON this is done using either second-order trust-region optimizations [22] (energies, gradients and Hessians are calculated) or a variety of first-order methods [23] (only energies and gradients are calculated). These methods have been implemented for Hartree–Fock, Density Functional Theory, MCSCF and various Coupled-Cluster wave functions. For other non-variational wave functions—such as CI—the program can only do first-order geometry optimizations using a numerically calculated gradient. This will be invoked automatically by the program in case of a non-variational wave function. Some comments connected to such geometry optimizations are collected in Sec. 7.3.

For historical reasons, DALTON actually contains two different modules for exploring potential energy surfaces: *WALK which is a pure second-order module and *OPTIMIZE which contains both first- and second-order methods. While there is a lot of overlap between the two modules, certain calculations can only be done using one or the other of the two. The main strength of *WALK is it's robustness, while *OPTIMIZE focuses more on speed and efficiency. For regular optimizations of minima and transition states, the recommended module is *OPTIMIZE.

One of the strengths of the DALTON program package is the stable algorithms for locating first-order transition states. As described below, this can done by one of three algorithms in *WALK: by trust-region second-order image surface minimization [24], by gradient extremal walks [25], or by following a specific mode [26] (all requiring the calculation of the Hessian at every point). These options will be discussed in more detail below. The *OPTIMIZE module also contains the stable second-order trust-region image surface minimization (using analytical or approximate Hessians), as well as a partitioned rational function method [27].

Another feature of the DALTON program system is the options for calculating essen-

tial information about the potential energy surface so that subsequent molecular dynamics analysis can be made. There are two options for doing this in the program: One can either follow an Intrinsic Reaction Coordinate (IRC) [25], or solve Newton's equations for the nuclei under the potential put up by the electrons [28]. Whereas the first option can be used for modeling the immediate surroundings of a molecular reaction path, the second will give a precise description of *one* molecular trajectory. Chemical reactions may thus be monitored in time from an initial set of starting conditions. The program will automatically generate a complete description of the energy distribution into internal energies and relative translational energies. By calculating a large number of trajectories, a more complete description of the chemical reaction may be obtained [29].

Please note that for geometry optimizations using the Self-Consistent Reaction Field model, certain restrictions apply, as discussed in Sec. 15.2.1.

7.1 Locating stationary points

7.1.1 Equilibrium geometries

> **Reference literature:**
>
> 2.nd-order methods: T.U.Helgaker, J.Almlöf, H.J.Aa.Jensen, and P.Jørgensen. *J.Chem.Phys.*, **84**, 6266, (1986).
>
> 1.st-order methods: V.Bakken, and T.Helgaker, *J.Chem.Phys.*, **117**, 9160 (2002).
>
> Excited-state optimization: A.Cesar, H.Ågren, T.Helgaker, P.Jørgensen, and H.J.Aa.Jensen. *J.Chem.Phys.*, **95**, 5906, (1991).
>
> Solvent Hessian: P.-O.Åstrand, K.V.Mikkelsen, K.Ruud and T.Helgaker. *J.Phys.Chem.*, **100**, 19771, (1996).

A short input file for the location of a molecular geometry corresponding to minimum molecular energy (using a second-order optimization method) and performing a vibrational analysis at the stationary point, as well as determining the nuclear shielding constants at the optimized geometry, will look like (assuming a Hartree–Fock wave function):

```
**DALTON INPUT
.OPTIMIZE
*OPTIMIZE
.2NDORD
**WAVE FUNCTIONS
.HF
**PROPERTIES
```

```
.VIBANA
.SHIELD
**END OF DALTON INPUT
```

The keyword `.OPTIMIZE` by default signals a search for a minimum, that is, a stationary point on the molecular surface with Hessian index 0. With the `*OPTIMIZE` module the second-order method has to be explicitly requested (`.2NDORD`) [22], as first-order methods are the default. At the final, optimized geometry the input in the `**FINAL` section requests a vibrational analysis and the evaluation of nuclear shieldings.

In second-order methods the Hessian is calculated at every geometry. The analytical Hessian naturally gives a better description of the potential energy surface than the approximate Hessians of first-order methods, and fewer steps will usually be needed to reach the minimum [30]. However, the price one has to pay for this, is that each iteration uses significantly more CPU time. Another advantage of second-order methods, is that the program will automatically break the symmetry of the molecule, if needed, in order to reach a minimum (unless the user specifies that symmetry should not be broken). If one wants to minimize a water molecule starting from a linear geometry and using molecular symmetry, a first-order method will "happily" yield a linear geometry with optimized bond lengths, whereas the second-order method will correctly decide that symmetry should be reduced to C_{2v} and minimize the energy of the molecule. The bottom-line is, however, that while second-order methods definitely are the more robust, they will generally be outperformed by the much more time-efficient first-order methods.

If you are convinced that the symmetry of your starting geometry is the correct, you may take advantage of the fact there will only be forces in the totally symmetric modes of the molecules, and thus only evaluate the Hessian in the totally symmetric Cartesian symmetry coordinates. This can be achieved with the input:

```
**DALTON INPUT
.WALK
.TOTSYM
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VIBANA
.SHIELD
**END OF DALTON INPUT
```

In case a vibrational analysis has been requested at the optimized geometry, the program will recalculate the complete molecular Hessian at the optimized geometry in

order to be able to evaluate all frequencies. Note that only the `*WALK` module, indicated by the keyword `.WALK`, supports the `.TOTSYM` keyword. We also note that any requests for static polarizabilities, Cioslowski population analysis, dipole gradients or vibrational circular dichroism (VCD) will be ignored as all the necessary property vectors will not be available. However, at the final optimized geometry, the full Hessian will be evaluated, as may then also these properties.

For direct calculations—where the two-electron integrals are not stored on disc—the cost of a second-order geometry optimization algorithm may become prohibitively expensive, despite the rather few iterations normally needed to converge the geometry. In these cases, it may be advantageous to use first-order geometry optimization routines as illustrated in the following input:

```
**DALTON INPUT
.OPTIMIZE
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VIBANA
.SHIELD
**END OF DALTON INPUT
```

This will use the default method of `.OPTIMIZE`: a first-order optimization [23] in redundant internal coordinates [31, 32, 30] using the BFGS Hessian update. This means that only the energy and the gradient is calculated in each iteration, and an approximate Hessian is obtained using the gradients and the displacements. The initial Hessian is diagonal in the redundant internal coordinates [33]. To obtain the properties at the optimized geometry, the Hessian has to be calculated. By looking at the vibrational frequencies one can then verify that a true minimum has been reached (as all frequencies should be real, corresponding to a positive definite Hessian).

Several first-order methods have been implemented in DALTON, the recommended method being the Broyden-Fletcher-Goldfarb-Shanno (BFGS) Hessian updating scheme. Though other updates may perform better in certain cases, the BFGS update generally seems to be the most reliable and thus is the default method for minimization [30].

Without the calculation of the Hessian at the optimized geometry, one can never be sure that the geometry is indeed a minimum, and this is the main problem with first-order methods.

Both first- and second-order methods are based on trust-region optimization. The trust-region is the region of the potential energy surface where our quadratic model (based on an analytical or approximate Hessian) gives a good representation of the true surface.

This region is given the shape of a hypersphere. In the trust-region based optimization algorithm, the radius of the trust-region is automatically updated during the optimization by a feedback mechanism. Occasionally, however, the potential surface may show locally large deviations from quadratic form. This will result in a large disagreement between the predicted energy change and the energy calculated at the new geometry. If this deviation is larger than a given threshold, the step will be reduced through a decrease in the trust radius and a simple line search will be performed. A quadratic function is fitted to the rejected energy and the previous energy and gradient. The minimum of this function is then used as the new step, and this process may be repeated.

Another consideration concerning optimizations, is the choice of coordinate system. Second-order methods does not seem to be very sensitive to this, and the `*WALK` module thus only uses Cartesian coordinates. However, the coordinate system may be crucial to the performance of first-order methods, especially when starting from approximate Hessians. `*OPTIMIZE` provides the normal Cartesian coordinates and redundant internal coordinates, the latter being highly recommended when using first-order methods. Redundant internal coordinates consists of all bond lengths, angles and dihedral angles in a molecule, and the redundancies are removed through projection. The default for second-order methods is Cartesian coordinates, while redundant internal coordinates is the default for first-order methods, but it is possible to override this with the keywords `.REDINT` and `.CARTES` respectively.

The `*OPTIMIZE` module decides when a minimum is reached through a set of convergence criteria. These will be set automatically depending on the accuracy of the wave function using three different criteria, namely the change in energy between two iterations, the norm of the molecular gradient and the norm of the step. The convergence threshold for the change in energy is the maximum of $1.0 \cdot 10^{-6}$ Hartree and two times the convergence threshold for the wave function, whereas for the norm of the gradient and step it is the maximum of $1.0 \cdot 10^{-5}$ a.u. and two times the convergence threshold for the wave function. Two out of these three criteria have to decrease below their threshold before the program declares convergence. These criteria ensure a good geometry suitable for vibrational analysis, but may be unnecessary tight if one only wants the energy.

The convergence criteria can be controlled through the input file. One should keep in mind that while second-order methods are rather insensitive to changes in these thresholds due to their quadratic convergence, first-order methods may run into severe trouble if the criteria are too close to the accuracy of the wave function and the gradient. The quickest way to get somewhat looser convergence criteria, is to use the keyword `.BAKER` enforcing the convergence criteria of Baker[34] (see the Reference Manual for details). This will normally give energies converged to within $1.0 \cdot 10^{-6}$ Hartree.

In case of non-variational wave functions, where a numerical gradient is used, the

threshold for convergence is reset to $1.0 \cdot 10^{-4}$ a.u., due to possible numerical instabilities occurring in the finite difference methods employed to determine the molecular gradient.

We also note that for the optimization of excited states where the state of interest is not the lowest of its symmetry, most of the parameters controlling the trust-region optimization method will be reset to much more conservative values, as one in such calculations will loose the possibility of back-stepping during the wave function optimization, as well as to ensure that the start wave function at the new geometry is within the local region of the optimized wave function.

Calculations involving large basis sets may be very time-consuming, and it becomes very important to start the optimization from a decent geometry. One way to solve this is through preoptimization, that is, one or more smaller basis sets are first used to optimize the molecule. *OPTIMIZE supports the use of up to ten different basis sets for preoptimization. This may also be used as a very effective way to get the optimized energy of a molecule for a series of basis sets. One may also want to calculate the energy of a molecule using a large basis set at the optimized geometry of a smaller basis set, and this is also supported in DALTON, but limited to only one single-point basis set. Both these features are illustrated in the input file below:

```
**DALTON INPUT
.OPTIMIZE
*OPTIMIZE
.PREOPT
2
STO-3G
4-31G
.SP BAS
d-aug-cc-pVTZ
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

As one can see, two small basis sets have been chosen for preoptimization. Note that they will be used in the order they appear in the input file, so one should sort the sets and put the smaller one at the top for maximum efficiency. The "main" basis set, that is, the one that will be used for the final optimized geometry, is specified in the file MOLECULE.INP as usual. After the last optimization, the energy of the molecule will be calculated at the final geometry with the d-aug-cc-pVTZ basis. Please note that these features will work only when using basis sets from the basis set library.

It is possible to control the optimization procedure more closely by giving the program instructions on how to do the different parts of the calculation. Below we have listed

all the modules that may affect the geometry optimization, as well as a short description of which part of the calculation that module controls. The reader is referred to the section describing each module for a closer description of the different options. These sections may, with the exception of the `*OPTIMIZE` and `*WALK` section, be specified in any of the modules `**START`, `**EACH STEP` and `**PROPERTIES`. The `*OPTIMIZE` and `*WALK` is to be placed in the `**DALTON` module, and will apply to the entire calculation.

`*GEOANA` Describes what kind of information about the molecular geometry is to be printed.

`*GETSGY` Controls the set up of the right-hand sides (gradient terms).

`*NUCREP` Controls the calculation of the nuclear contributions.

`*ONEINT` Controls the calculation of one-electron contributions.

`*OPTIMIZE` Controls the first- and second-order optimization methods (both minima and transition states).

`*RELAX` Controls the adding of solution and right-hand side vectors into relaxation contributions.

`*REORT` Controls the calculation of reorthonormalization terms.

`*RESPON` Controls the solution of the geometric response equations.

`*TROINV` Controls the use of translation and rotational invariance.

`*TWOEXP` Controls the calculation of two-electron expectation values.

`*VIBANA` Set up the vibrational and rotational analysis of the molecule.

`*WALK` Controls the walk (see also "Locating transition states" and "Doing Dynamical walks").

7.1.2 Transition states using the image method

> **Reference literature:**
>   T.Helgaker. *Chem.Phys.Lett.*, **182**, 503, (1991).

Transition states are found as saddle points on the potential energy surface. The simplest way of locating a first-order transition state, which are the chemically most interesting ones, is to use the trust-region image minimization method described in Ref. [24]. Such geometry optimizations may be considered as a special case of walks on second-order surfaces, and can be done using either the `*WALK` module or the `*OPTIMIZE` module. Just

like for minimization the former uses a pure second-order method (analytical Hessians calculated at every geometry), while the latter gives you a choice of first- and second-order methods or combinations of the two.

A second-order optimization of a transition state can be requested by either adding `.SADDLE` and `.2NDORD` in the `*OPTIMIZE` section:

```
**DALTON INPUT
.OPTIMIZE
*OPTIMIZE
.SADDLE
.2NDORD
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

or by adding the keyword `.IMAGE` in the `*WALK` section:

```
**DALTON INPUT
.WALK
*WALK
.IMAGE
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

Any property may of course be specified at all stages of the optimization in the same fashion as for geometry minimizations.

The principle behind the trust-region image minimization is simple. A first-order transition state is characterized by having one negative Hessian eigenvalue. By reversing the sign of this eigenvalue, we have taken the mirror image of our potential surface along the associated mode, thus turning our problem into an ordinary minimization problem. A global one-to-one correspondence between the image surface and our potential energy surface is only valid for a second-order surface, but in general the lack of a global one-to-one correspondence seldom gives any problems.

The advantage of the trust-region image optimization as compared to for instance following gradient extremals lies mainly in the fact that we may take advantage of well-known techniques for minimization. In addition, the method does not need to be started at a stationary point of the potential surface which is necessary when following a gradient extremal (in fact, when using `*OPTIMIZE` the starting geometry should *not* be a minimum). We have so far not experienced that the trust-region image optimization fails to locate

a first order transition state, even though this is by no means globally guaranteed from the approach. Note that the first-order saddle-points normally obtained using the image method starting at a minimum, often corresponds to conformational transition states, and thus not necessarily to the chemically most interesting transition states.

There are two approaches for locating several first-order saddle points with the trust-region image optimization. One may take advantage of the fact the image method is not dependent upon starting at a stationary point, and thus start the image minimization from several different geometries, and thus hopefully ending up at different first-order saddle points, as the lowest eigenmode at different regions of the potential energy surface may lead to different transition states.

The other approach is to request that not the lowest mode, but some other eigenmode is to be inverted. This can be achieved by explicitly giving the mode which is to be inverted through the keyword .MODE. This keyword is the same in both the *WALK and the *OPTIMIZE module. However, one should keep in mind that there will be crossings where a given mode will switch from the chosen mode to a lower mode. However, what will happen in these crossing points cannot be predicted in advance. Thus, for such investigations, the gradient extremal approach may prove equally well suited. Let us give an example of an input for a trust-region image optimization where the third mode is inverted:

```
**DALTON INPUT
.WALK
*WALK
.IMAGE
.MODE
 3
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

7.1.3 Transition states using first-order methods

While second-order methods are robust, we have already pointed out in section 7.1.1 that Hessians might be expensive to compute. The *OPTIMIZE module therefore provides first-order methods for locating transition states, where approximate rather than exact Hessians are used. The step control method is, however, the same trust-region image optimization.

When locating transition states it is important to have a good description of the mode that should be maximized (*i.e.* have a negative eigenvalue). It is therefore not recommended to start off with a simple model Hessian, but rather to calculate the initial Hessian analytically. Alternatively it can be calculated using a smaller basis set/cheaper

wave function and then be read in. The minimal input for a transition state optimization using `*OPTIMIZE` is:

```
**DALTON INPUT
.OPTIMIZE
*OPTIMIZE
.SADDLE
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

This will calculate the Hessian at the initial geometry, then update it using Bofill's update [35] (the BFGS update is not suitable since it tends towards a positive definite Hessian). As for minimizations, redundant internal coordinates are used by default for first-order methods.

It is highly recommended that a Hessian calculation/vibrational analysis be performed once a stationary point has been found, to verify that it's actually a first-order transition state. There should be one and only one negative eigenvalue/imaginary frequency.

If no analytical second derivatives (Hessians) are available, it is still possible to attempt a saddle point optimization by starting from a model Hessian indicated by the keyword `.INIMOD`:

```
**DALTON INPUT
.OPTIMIZE
*OPTIMIZE
.INIMOD
.SADDLE
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

Provided the starting geometry is reasonably near the transition state, such optimization will usually converge correctly. If not, it is usually a good idea to start from different geometries and also to try to follow different Hessian modes, as described in section 7.1.2 (through the `.MODE` keyword).

7.1.4 Transition states following a gradient extremal

> **Reference literature:**
>
> P.Jørgensen, H.J.Aa.Jensen, and T.Helgaker. *Theor.Chem.Acta*, **73**, 55, (1988).

A gradient extremal is defined as a locus of points in the contour space where the gradient is extremal [25]. These gradient extremals connect stationary points on a molecular potential energy surface and are locally characterized by requiring that the molecular gradient is an eigenvector of the mass-weighted molecular Hessian at each point on the line. From a stationary point there will be gradient extremals leaving in all normal coordinate directions, and stationary points on a molecular potential surface may thus be characterized by following these gradient extremals. The implementation of this approach in DALTON is described in Ref. [25].

Before discussing more closely which keywords are of importance in such a calculation and how they are to be used, we give an example of a typical gradient extremal input in a search for a first-order transition state along the second-lowest mode of a mono-deuterated ethane molecule:

```
**DALTON INPUT
.WALK
*WALK
.GRDEXT
.INDEX
 1
.MODE
 2
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

The request for a gradient extremal calculation is controlled by the `.GRDEXT`, and in this example we have chosen to follow the second-lowest mode, as specified by the `.MODE` keyword.

As the calculation of the gradient extremal uses mass-weighted coordinates, it is recommended to specify the isotopic constitution of the molecule. If none is specified, the most abundant isotope of each atom is used by default. The isotopic constitution of a molecule is given in the `MOLECULE.INP` file as described in Chapter 23.

A requirement in a gradient extremal calculation is that the calculation is started on a gradient extremal. In practice this is most conveniently ensured by starting at a stationary point, a minimum or a transition state. The index of the critical point — that is, the number of negative Hessian eigenvalues — sought, need to be specified (by the keyword `.INDEX`), as the calculation would otherwise continue until a critical point with index zero (corresponding to a minimum), is found.

7.1.5   Level-shifted mode-following

> **Reference literature:**
>
> C.J.Cerjan, and W.H.Miller. *J.Chem.Phys.*, **75**, 2800, (1981).
>
> H.J.Aa.Jensen, P.Jørgensen, and T.Helgaker. *J.Chem.Phys.*, **85**, 3917, (1986).

The input needed for doing a level-shifted mode following is very similar to the input for following a gradient extremal, and the keyword that is needed in order to invoke this kind of calculation is `.MODFOL`. As for gradient extremals, we need to specify which mode we follow. However, a mode following does not use mass-weighted molecular coordinates as default, and isotopic composition of the molecule is therefore not needed. Note, however, that mass-weighted coordinates can be requested through the keyword `.MASSES` as described in the input section for the `*WALK` module. A typical input following the third mode will thus look like:

```
**DALTON INPUT
.WALK
*WALK
.MODFOL
.INDEX
 1
.MODE
 3
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

The level-shifted mode-following uses an algorithm similar to the one used in the ordinary geometry optimization of a molecule, but whereas one in minimizations chooses a step so that the level shift parameter is less than the lowest eigenvalue of the molecular Hessian, this level shift parameter is chosen to be in-between the eigenvalues λ_{t-1} and λ_t if we are following mode number t. This approach was pioneered by Cerjan and Miller [36], and is also described in Ref. [26]. As for the gradient extremal approach, higher-order transition states can be requested through the use of the keyword `.INDEX`.

Note that it may often be necessary to start the mode-following calculation by stepping out of the stationary point along the mode of interest using the keyword `.EIGEN` in the `.WALK` module. We refer to the reference manual for a further description of this option.

The index of the critical point—that is, the number of negative Hessian eigenvalues—sought, need to be specified (by the keyword `.INDEX`), as the calculation would otherwise continue until a critical point with index zero (corresponding to a minimum), is found.

7.2 Trajectories and Dynamics

7.2.1 Intrinsic reaction coordinates

Reference literature:

K.Fukui. *Acc.Chem.Res.*, **14**, 363, (1981).

A tool that have proved valuable in the study of molecular dynamics is the use of steepest descent-based algorithms for following a molecular reaction from a transition state towards a minimum. One of the most successful ways of doing this is the Intrinsic Reaction Coordinate (IRC) approach [37]. The IRC is calculated by taking small steps along the negative gradient in a mass-weighted coordinate system. In DALTON, the size of the step is adjusted with the use of the trust-region based algorithm. In order to get a sufficiently accurate potential energy surface, rather small steps must be taken, and the default trust radius is thus reset to 0.020 when an IRC calculation is being done.

In many respects the input to an IRC calculation is very similar to the input for a trust-region image optimization, and a typical input would look like:

```
**DALTON INPUT
.WALK
.MAX IT
 150
*WALK
.IRC
 1
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

Most of this input should now be self-explanatory. The request for an Intrinsic Reaction Coordinate calculation is done by using the keyword `.IRC`. On the next line there is a positive or negative integer indicating in which direction the reaction should proceed. It is, however, not possible to determine in advance which reaction path a given sign is connected to, and the calculation should therefore always be checked after a few iterations in order to ensure that the reaction proceeds in the correct direction. If not, the calculation

should be stopped and started from the transition state again with a different sign for the integer specified after the `.IRC` keyword.

As the IRC is defined with respect to mass-weighted coordinates, care has to be taken in order to specify the correct isotopic substitution of the molecule. The specification of the isotopic constitution of a molecule is given in the `MOLECULE.INP` file, as described in Chapter 23.

Due to the small steps that must be used in a calculation of an IRC, such a calculation may require a large number of iterations, and it may thus be necessary to increase the maximum number of iterations that can be taken. This can be done by the keyword `.MAX IT` in the `**DALTON` input section. Default value for this parameter is 20 iterations.

All the information about the Intrinsic Reaction Coordinate will be collected in a file called `DALTON.IRC`. If a calculation stops because it has reached the maximum number of iterations, it may be restarted from that point, and the new information about the IRC will be added to the old `DALTON.IRC` file. This also implies that if a calculation is restarted from the beginning (because it went in the wrong direction) the `DALTON.IRC` file *must* be removed first. Thus it may often be useful to take a backup of the `DALTON.IRC` during the calculation of the IRC.

Finally, some comments on the interest of a calculation of IRCs. Whereas it will give results that mimic the behavior of what is considered to be a good description of the reaction pathway of a molecular reaction, it does not include any dynamical aspects of the reaction. There exists several models for approximating local regions of a molecular potential energy surface from the results of an IRC calculation, and from the potential energy surface valley, the dynamics of a chemical reaction may be mimicked. However, DALTON gives another, more direct, opportunity for studying molecular dynamics through dynamic walks as described in more detail in the next section.

7.2.2 Doing a dynamical walk

> **Reference literature:**
>
> T.Helgaker, E.Uggerud, H.J.Aa.Jensen. *Chem.Phys.Lett.*, **173**, 145, (1990).

The theory behind the "direct dynamics" as implemented in DALTON is described in Ref. [28]. The main idea behind this approach is that Newton's equations of motion for the nuclei are integrated in the presence of the quantum mechanical potential set up by the electrons. Thus one may follow a molecular reaction from a given starting point (usually a transition state) as it would behave if the nuclei could be treated exactly as classical particles. One should also keep in mind that the Hamiltonian used is constructed within the framework of the Born-Oppenheimer approximation, which may turn out not to

be a good approximation at given points during the reaction. Furthermore, the calculation describes the way molecules with a predefined orientation and momentum will react. Thus the trajectory obtained is only one of a large number of possible trajectories depending on the initial state of the molecule.

The necessary input in order to do a dynamical walk of for instance protonated formaldehyde would look like:

```
**DALTON INPUT
.WALK
.MAX IT
 200
*WALK
.DYNAMI
.FRAGME
 5
 1 1 1 2 2
.MOMENT
 1
1 -.00001
.MODE
 1
**WAVE FUNCTIONS
.HF
**END OF DALTON INPUT
```

The walk is specified to be a dynamic walk through the keyword `.DYNAMI`. The starting trust radius will in dynamical calculations be changed to a new default value of 0.005.

The keyword `.FRAGME` dictates which atoms belong to which molecular fragment. In this particular case, we assume that two protons leave the protonated formaldehyde as a hydrogen molecule, and that the leaving hydrogen atoms are the last atoms of the MOLECULE input file. This partitioning is mainly needed in order to get proper values for the relative translational energy between the two fragments, as well as for deciding how much of the energy has been distributed into internal degrees of freedom.

The default isotopic substitution is that the most abundant isotopes are to be used in the calculation. Isotopic substitution is important as the masses of the nuclei enters when Newton's equations of motion are integrated. The specification of the isotopic constitution of the molecule is given in the `MOLECULE.INP` file, as described in Chapter 23.

In this calculation we start the calculation at a transition state, and in order to get the reaction started we need to give the molecule a slight push. This is achieved by the

keyword `.MOMENT`. In the next line the user then specifies the number of modes in which there is an initial momentum, followed by lines containing pairs of numbers, of which the first specifies the mode, and the second the momentum in this mode. There must be as many pairs of modes and momenta as specified in the line after the `.MOMENT` keyword. It is impossible to predict in advance which way the reaction will proceed, and the calculation should be checked after a few iterations, in order to ensure that it proceeds in the right direction. If not, the calculation should be started from the transition state again with a different sign on the initial momentum. The `DALTON.TRJ` must also be removed as discussed below.

It is in principle possible to start a calculation from any point on a molecular potential energy surface, and in cases where these starting points do not correspond to a stationary point, `.MOMENT` may be skipped, as there exist a downward slope (in other words, an attractive force) driving the molecule(s) in a specific direction. One may of course also start the molecule with a given initial momentum in different energy modes.

During the dynamical calculation, care has also to be taken in order to ensure that the steps taken are not too long. If this occurs, the initial trust radius and/or the trust radius increment should be reduced by the keyword `.TRUST`. In the DALTON output one will find "Accumulated kinetic energy since start", and this property will be calculated in two ways: From conservation of the total energy, and from integrated momenta. If the difference between these numbers is larger than approximately 1% , the calculation should be stopped and the starting trust radius be decreased and the calculation restarted from the starting point again after removal of the `DALTON.TRJ`-file.

The calculation of dynamical walks may take from about 70 to 1200 iterations (as a general rule) and one must therefore adjust the maximum number of iterations allowed. This is done by the `.MAX IT` keyword. In the present example the maximum number of iterations have been reset to 200. If the calculation cannot be closely monitored, it is recommended not to set the maximum number of iterations too high, and rather restart the calculation if this turns out to be necessary. This can be accomplished by specifying the iteration at which the calculation will restarted by the keyword `.ITERAT` in the `**DALTON` input module.

The calculation should be stopped (at least for ordinary hydrogen elimination reactions) when the "Relative velocity" starts to decrease, as this indicates that the molecules are so far apart that basis set superposition errors become apparent. We will below return to how one then calculates translational energy release of the reaction.

During the whole calculation, a file `DALTON.TRJ` is updated. This file contains information from the entire dynamic walk. If a walk is restarted from a given point, the new information will be appended to the old `DALTON.TRJ`-file. Note that this also implies that if you need to restart the calculation from the beginning (because the reaction went the wrong

way or because of a too large trust radius), the `DALTON.TRJ`-file *must* be removed. Thus, it may often be advisable to take a backup of this file in certain parts of the calculation. As this file contains all the information about the dynamical walk, this file can be used to generate a video-sequence of the molecular reaction along this specific trajectory with the correct time-scaling [38].

7.2.3   Calculating relative translational energy release

It is often of interest to calculate the relative translational energy release in a given reaction, as this can be compared to experimental values determined from *e.g.* mass spectrometry [28]. Although this quantity is printed in the output from DALTON in the entire dynamical walk, the relative translational energy release should be calculated, due to basis set superposition errors and vibrational and rotational excitation in the departing molecular fragments, in the way described here.

The geometry of the last iterations for which relative translational energy release is known, is used as a starting point for minimizing the two molecular fragments as described in Section 7.1.1. As a check of this minimization one might also minimize the two molecular fragments separately, and check this total energy against the energy obtained when minimizing the molecular supersystem. The energies should be almost identical, but small differences due to basis set superposition errors may be noticeable.

The barrier height can then be calculated by subtracting the energy of the molecule at the transition state and the energy for the separated molecular fragments, or an experimentally determined barrier height may be used. The relative translational energy release may then be obtained by dividing the translational energy release from the last DALTON iteration by the barrier height. This number will not be identical to the number printed in the DALTON-output, because of the different vibrational and rotational state of the molecule in the final iteration point as compared to the minimized structure.

7.3   Geometry optimization using non-variational wave functions

DALTON does not have any support for the calculation of molecular gradients and Hessians for the non-variational wave functions CI and NEVPT2. However, in order to exploit the facilities of the first-order geometry optimization routines in DALTON, a numerical gradient based on energies will be calculated if a geometry optimization is invoked for a non-variational wave function. As a simple example, to optimize the MP2 geometry of a molecule using numerical gradients[1], the only input needed is

---

[1]Note that MP2, CCSD, and CCSD(T) analytical gradients are available through the CC module

```
**DALTON INPUT
.OPTIMIZE
**WAVE FUNCTIONS
.HF
.MP2
**END OF DALTON INPUT
```

The size of the displacements used during the evaluation of the numerical gradient can be controlled through the keyword `.DISPLA` in the `*OPTIMI` input module. Default value is $1.0 \cdot 10^{-3}$ a.u. By default, the threshold for convergence of the geometry will be changed because of estimated inaccuracies in the numerical gradients. However, if the threshold for convergence is altered manually, this user supplied threshold for convergence will be used also in geometry optimizations using numerical gradients. Note that due to the possibility of larger numerical errors in the gradient, too tight convergence criteria for an optimized geometry may make it difficult for the program to obtain a converged geometry.

Chapter 8

Molecular vibrations

In this chapter we discuss properties related to the vibrational motions of a molecule. This includes vibrational frequencies and the associated infrared (IR) and Raman intensities.

8.1   Vibrational frequencies

The calculation of vibrational frequencies are controlled by the keyword `.VIBANA`. Thus, in order to calculate the vibrational frequencies of a molecule, all that is is needed is the input:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VIBANA
**END OF DALTON INPUT
```

This keyword will, in addition to calculating the molecular frequencies, also calculate the zero-point vibrational energy corrections and vibrational and rotational partition functions at selected temperatures.

DALTON evaluates the molecular Hessian in Cartesian coordinates, and the vibrational frequencies of any isotopically substituted species may therefore easily be obtained on the basis of the full Hessian. Thus, if we would like to calculate the vibrational frequencies of isotopically substituted molecules, this may be obtained through an input like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
```

```
.HF
**PROPERITES
.VIBANA
*VIBANA
.ISOTOP
   2   5
 1 2 1 1 1
 2 1 1 1 1
**END OF DALTON INPUT
```

The keyword `.ISOTOP` in the `*VIBANA` input module indicates that more than only the isotopic species containing the most abundant isotopes are to be calculated, which will always be calculated. The numbers on the second line denotes the number of isotopically substituted species that is requested and the number of atoms in the system. The following lines then lists the isotopic constitution of each of these species. 1 corresponds to the most abundant isotope, 2 corresponds to the second most abundant isotope and so on. The isotopic substitution have to be given for all atoms in the molecule (not only the symmetry independent), and the above input could for instance correspond to a methane molecule, with the isotopic species CH_3D and $^{13}CH_4$.

As the isotopic substitution of all atoms in the molecule has to be specified, let us mention the way symmetry-dependent atoms will be generated. The atoms will be grouped in symmetry-dependent atom blocks. The specified symmetry-independent atom will be the first of this block, and the symmetry-dependent atoms will be generated according to the order of the symmetry elements. Thus, assuming D_{2h} symmetry with symmetry generating elements `X  Y  Z`, the atoms generated will come in the order `X`, `Y`, `XY`, `Z`, `XZ`, `YZ`, and `XYZ`.

8.2 Infrared (IR) intensities

> **Reference literature:**
>
> R.D.Amos. *Chem.Phys.Lett.*, **108**, 185, (1984).
>
> T.U.Helgaker, H.J.Aa.Jensen, and P.Jørgensen. *J.Chem.Phys.*, **84**, 6280, (1986).

The evaluation of infrared intensities requires the calculation of the dipole gradients (also known as Atomic Polar Tensors (APTs)). Thus, by combining the calculation of vibrational frequencies with the calculation of dipole gradients, IR intensities will be obtained. Such an input may look like:

```
**DALTON INPUT
```

```
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VIBANA
.DIPGRA
*VIBANA
.ISOTOP
   2   5
 1 2 1 1 1
 2 1 1 1 1
**END OF DALTON INPUT
```

The keyword `.DIPGRA` invokes the calculation of the dipole gradients.

8.3 Dipole-gradient based population analysis

> **Reference literature:**
>
> J.Cioslowski. *J.Am.Chem.Soc.*, **111**, 8333, (1989).
>
> T.U.Helgaker, H.J.Aa.Jensen, and P.Jørgensen. *J.Chem.Phys.*, **84**, 6280, (1986).

As dipole gradients are readily available in the DALTON program, the population analysis basis on the Atomic Polar Tensor as suggested by Cioslowski [39, 40] can be obtained from an input like

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.POPANA
*END OF DALTON
```

This population analysis is of course significantly more expensive than the ordinary Mulliken population analysis obtainable directly from the molecular wave functions through an input like

```
**DALTON INPUT
```

```
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.HF
*POPULATION ANALYSIS
.MULLIKEN
*END OF DALTON
```

8.4 Raman intensities

> **Reference literature:**
>
> T.Helgaker, K.Ruud, K.L.Bak, P.Jørgensen, and J.Olsen. *Faraday Discuss.*, **99**, 165, (1994).

Calculating Raman intensities is by no means a trivial task, and because of the computational cost of such calculations, there are therefore few theoretical investigations of basis set requirements and correlation effects on calculated Raman intensities. The Raman intensities calculated are the ones obtained within the Placzek approximation [41], and the implementation is described in Ref. [42].

The Raman intensity is the differentiated frequency-dependent polarizability with respect to nuclear displacements. As it is a third derivative depending on the nuclear positions through the basis set, numerical differentiation of the polarizability with respect to nuclear coordinates is necessary.

The input looks very similar to the input needed for the calculation of Raman optical activity described in Section 11.4

```
**DALTON INPUT
.WALK
*WALK
.NUMERI
**WAVE FUNCTIONS
.HF
*SCF INPUT
.THRESH
1.0D-8
**START
.RAMAN
*ABALNR
.THRESH
1.0D-7
```

```
.FREQUE
     2
0.0 0.09321471
**EACH STEP
.RAMAN
*ABALNR
.THRESH
1.0D-7
.FREQUE
     2
0.0 0.09321471
**PROPERTIES
.RAMAN
.VIBANA
*RESPONSE
.THRESH
1.0D-6
*ABALNR
.THRESH
1.0D-7
.FREQUE
     2
0.0 0.09321471
*VIBANA
.PRINT
 1
.ISOTOP
   1   5
 1 1 1 2 3
**END OF DALTON INPUT
```

The keyword `.RAMAN` in the general input module indicates that a frequency-dependent polarizability calculation is to be done. The keyword `.RAMAN` indicates that we are only interested in the Raman intensities and depolarization ratios. Note that these parameters are also obtainable by using the keyword `.VROA`. In this calculation we calculate the Raman intensities for two frequencies, the static case and a frequency of the incident light corresponding to a laser of wavelength 488.8 nm.

Due to the numerical differentiation that is done, the threshold for the iterative solution of the response equations are by default 10^{-7}, in order to get Raman intensities

that are numerically stable to one decimal digit.

In the `*WALK` input module we have specified that the walk is a numerical differentiation. This will automatically turn off the calculation of the geometric Hessian, putting limitations on what kind of properties that may be calculated at the same time as Raman intensities. Because the Hessian is not calculated, there will not be any prediction of the energy at the new point.

It should also be noted that in a numerical differentiation, the program will step plus and minus one displacement unit along each Cartesian coordinate of all nuclei, as well as calculating the property at the reference geometry. Thus, for a molecule with N atoms the properties will need to be calculated in a total of $2*3*N + 1$ points, which for a molecule with five atoms will amount to 31 points. The default maximum number of steps in DALTON is 20. However, in numerical differentiation calculations, the number of iterations will always be reset (if there are more than 20 steps that need to be taken) to $6N+1$, as it is assumed that the user always wants the calculation to complete correctly. The maximum number of allowed iterations can be manually set by adding the keyword `.MAX IT` in the `**DALTON` input module.

The default step length in the numerical differentiation is $1.0 \cdot 10^{-4}$ a.u., and this step length may be adjusted by the keyword `.DISPLA` in the `*WALK` input module. The steps are taken in the Cartesian directions and not along normal modes. This enables us to study the Raman intensities of a large number of isotopically substituted molecules at once. This is done in the `**PROPERTIES` input section, where we have requested one isotopically substituted species in addition to the isotopic species containing the most abundant isotope of each element.

Chapter 9

Electric properties

This chapter describes the calculation of the different electric properties which have been implemented in the DALTON program system. These include the dipole moment, the quadrupole moment, the nuclear quadrupole interactions, and the static and frequency dependent polarizability. Note that a number of different electric properties may be obtained by use of the RESPONSE program if they can be expressed as a linear, quadratic or cubic response function. For the non-linear electric properties we refer to the chapter "Getting the property you want" (Chapter 12).

9.1 Dipole moment

The dipole moment of a molecule is always calculated if **PROPERTIES is requested, and no special input is needed in order to evaluate this property.

9.2 Quadrupole moment

The traceless molecular quadrupole moment, as defined by Buckingham [43], is calculated by using the keyword .QUADRU, and it can be requested from an input like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.QUADRU
**END OF DALTON INPUT
```

Note that both the electronic and nuclear contributions are always printed in the coordinate system chosen, that is, the tensors are not transformed to the principal axis system nor to the principal inertia system, as is often done in the literature.

The quadrupole moment is evaluated as an expectation value, and is thus fast to evaluate. This is noteworthy, because experimentally determined quadrupole moments obtained through microwave Zeeman experiments (see e.g. [44, 45]) are derived quantities and prone to errors, whereas the calculation of rotational g factors and magnetizability anisotropies (see Chapter 10)—obtainable from such experiments—are difficult to calculate accurately [46]. An input requesting a large number of the properties obtainable from microwave Zeeman experiments is (where we also include nuclear quadrupole coupling constants):

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.MAGNET
.MOLGFA
.QUADRU
.NQCC
**END OF DALTON INPUT
```

Note that the program prints the final molecular rotational g tensors in the principal inertia system, whereas this is not the case for the magnetizabilities and molecular quadrupole moment.

9.3 Nuclear quadrupole coupling constants

This property is the interaction between the nuclear quadrupole moment of a nucleus with spin greater or equal to 1, and the electric field gradient generated by the movement of the electron cloud around the nucleus. Quantum mechanically it is calculated as an expectation value of the electric field gradient at the nucleus, and it is obtained by the input:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
```

```
.NQCC
**END OF DALTON INPUT
```

It is noteworthy that due to it's dependence on the electronic environment close to the nucleus of interest, it puts strict demands on the basis set, similar to that needed for spin-spin coupling constants (Sec. 10.5). Electron correlation may also be of importance, see for instance Ref. [47].

9.4 Static and frequency dependent polarizabilities

Frequency-dependent polarizabilities is calculated from a set of linear response functions as described in Ref. [48]. In ABACUS the calculation of frequency-dependent linear response functions is requested through the keyword `.ALPHA` in the general input module to the property section. An input file requesting the calculation of the frequency dependent polarizability of a molecule may then be calculated using the following input:

```
**DALTON INPUT
.RUN PROPERTIES
**END OF DALTON INPUT
**WAVE FUNCTIONS
.HF
**PROPERTIES
.ALPHA
*ABALNR
.FREQUE
    2
0.0 0.09321471
**END OF DALTON INPUT
```

For a Second Order Polarization Propagator Approximation (SOPPA) [49, 50, 51, 52] calculation of frequency-dependent polarizabilities the additional keyword `.SOPPA` has to be specified in the `**PROPERTIES` input module and an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module. Similarly, for a SOPPA(CCSD) [53, 52] calculation of frequency dependent polarizabilities the additional keyword `.SOPPA(CCSD)` has to be specified in the `**PROPERTIES` input module and an CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the option `.SOPPA(CCSD)` in the `*CC INPUT` section.

The `*ABALNR` input section controls the calculation of the frequency-dependent linear response function. We must here specify at which frequencies the polarizability is to be

calculated. This is done with the keyword `.FREQUE`, and in this run the polarizability is to be evaluated at zero frequency (corresponding to the static polarizability) and at a frequency (in atomic units) corresponding to a incident laser beam of wavelength 488.8 nm.

There is also another way of calculating the static polarizability, and this is by using the keyword `.POLARI` in the `**PROPERTIES` input modules. Thus, if we only want to evaluate the static polarizability of a molecule, this may be achieved by the following input:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.POLARI
**END OF DALTON INPUT
```

Furthermore, the general RESPONSE program will also calculate the frequency-dependent polarizability as minus the linear response functions through the input

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
**RESPONSE
*LINEAR
.DIPLEN
**END OF DALTON INPUT
```

For further details about input for the response program, we refer to Chapter 12.

Chapter 10

Calculation of magnetic properties

This chapter describes the calculation of properties depending on magnetic fields, both as created by an external magnetic field as well as the magnetic field created by a nuclear magnetic moment. This includes the two contributions to the ordinary spin-Hamiltonian used in NMR, nuclear shieldings and indirect nuclear spin–spin couplings constants. We also describe the calculation of the magnetic analogue of the polarizability, the molecular magnetizability. This property is of importance in NMR experiments where the reference substance is placed in another tube than the sample. We also shortly describe two properties very closely related to the magnetizability and nuclear shieldings, respectively, the rotational g factor and the nuclear spin–rotation constants.

Three properties that in principle depend on the magnetic moments are not treated here, namely the properties associated with optical activity or, more precisely, with circular dichroism. These properties are Vibrational Circular Dichroism (VCD), Raman Optical activity (ROA) and Electronic Circular Dichroism (ECD) and these properties will be treated in Chapter 11. Another (magneto-)optical property, the $\mathcal{B}$ term of Magnetic Circular Dichroism (MCD) will be described in Chapter 12.

Gauge-origin independent nuclear shieldings, magnetizabilities and rotational g tensors are obtained through the use of London atomic orbitals, and the theory is presented in several references [54, 55, 56, 57]. Gauge-origin independent nuclear shieldings and magnetizabilities can also be obtained by using the the Continuous Transformation of the Origin of the Current Density method (CTOCD) approach [58, 59, 60]. In the present version of DALTON the CTOCD-DZ method is implemented and can be invoked by the keyword `.CTOCD` in the `**PROPERTIES` input module. More detailed information on CTOCD-DZ calculations can be found in section 10.9.1.

The indirect spin–spin couplings are calculated by using the triplet linear response function, as described in Ref. [61]. These are in principle equally simple to calculate with DALTON as nuclear shieldings and magnetizabilities. However, there are 10 contributions to

the spin–spin coupling constant from *each* nucleus.[1] Furthermore, the spin–spin coupling constants put severe requirements on the quality of the basis set as well as a proper treatment of correlation, making the evaluation of spin–spin coupling constants a time consuming task. Some notes about how this time can be reduced is given below.

Second Order Polarization Propagator Approximation (SOPPA) calculations [49, 50, 51, 62, 63] or SOPPA(CCSD) calculations [53, 62, 63] of the indirect spin–spin couplings, nuclear shieldings, magnetizabilities, rotational *g* tensors and the nuclear spin–rotation constants can be invoked by the additional keywords `.SOPPA` or `.SOPPA(CCSD)` in the `**PROPERTIES` input module. This requires for SOPPA that the MP2 energy was calculated by specifying the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for a SOPPA(CCSD) calculation the CCSD amplitudes have to be generated by specifying the keyword `.CC` in the `**WAVE FUNCTIONS` input module and `.SOPPA(CCSD)` in the `*CC INPUT` section. The use of London orbitals is automatically disabled in SOPPA calculations of the nuclear shieldings, magnetizabilities and rotational *g* tensors.

10.1 Magnetizabilities

> **Reference literature:**
>
> SCF magnetizabilities: K.Ruud, T.Helgaker, K.L.Bak, P.Jørgensen and H.J.Aa.Jensen. *J.Chem.Phys.*, **99**, 3847, (1993).
>
> MCSCF magnetizabilities: K.Ruud, T.Helgaker, K.L.Bak, P.Jørgensen, and J.Olsen. *Chem.Phys.*, **195**, 157, (1995).
>
> Solvent effects: K.V.Mikkelsen, P.Jørgensen, K.Ruud, and T.Helgaker. *J.Chem.Phys.*, **107**, 1170, (1997).
>
> CTOCD-DZ magnetizabilities: P.Lazzeretti, M.Malagoli and R.Zanasi. *Chem.Phys.Lett.*, **220**, 299, (1994)

The calculation of molecular magnetizabilities is invoked by the keyword `.MAGNET` in the `**PROPERTIES` input module. Thus a complete input file for the calculation of molecular magnetizabilities will look like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.MAGNET
```

---

[1] DALTON 2.0 now only calculates the symmetry-distinct contributions to the spin–dipole operator, that is—six instead of nine elements are calculated for this operator.

`**END OF DALTON INPUT`

This will invoke the calculation of molecular magnetizabilities using London Atomic Orbitals to ensure fast basis set convergence and gauge-origin independent results. The natural connection [64] is used in order to get numerically accurate results. By default the center of mass is chosen as gauge origin.

For a SOPPA or SOPPA(CCSD) calculation of molecular magnetizabilities, the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module. For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`. This will also automatically disable the use of London orbitals.

For a CTOCD-DZ calculation of molecular magnetizabilities, the additional keyword `.CTOCD` has to be specified in the `**PROPERTIES` input module. This will automatically disable the use of London orbitals. `.SOPPA` / `.SOPPA(CCSD)` and `.CTOCD` could be used together to get SOPPA / SOPPA(CCSD) molecular magnetizabilities using the CTOCD-DZ formalism. Information about suitable basis sets for CTOCD-DZ calculations can be found in the section 10.9.1.

The augmented cc-pVXZ basis sets of Dunning and coworkers [65, 66, 67, 68] have been shown to give to give excellent results for magnetizabilities [56, 69, 70], and these basis sets are obtainable from the basis set library.

Notice that a general print level of 2 or higher is needed in order to get the individual contributions (relaxation, one- and two-electron expectation values and so on) to the total magnetizability.

If more close control of the different parts of the calculation of the magnetizability is wanted, we refer the reader to the section describing the options available. The modules that controls the calculation of molecular magnetizabilities are:

`*EXPECT` Controls the calculation of one-electron expectation values contributing to the diamagnetic magnetizability.

`*GETSGY` Controls the set up of the right-hand sides (gradient terms) as well as the calculation of two-electron expectation values and reorthonormalization terms.

`*LINRES` Controls the solution of the magnetic response equations

`*RELAX` Controls the multiplication of solution and right-hand side vectors into relaxation contributions

10.2 Nuclear shielding constants

> **Reference literature:**
>
> K.Wolinski, J.F.Hinton, and P.Pulay. *J.Am.Chem.Soc.*, **112**, 8251, (1990)
>
> K.Ruud, T.Helgaker, R.Kobayashi, P.Jørgensen, K.L.Bak, and H.J.Aa.Jensen. *J.Chem.Phys.*, **100**, 8178, (1994).
>
> Solvent effects: K.V.Mikkelsen, P.Jørgensen, K.Ruud, and T.Helgaker. *J.Chem.Phys.*, **107**, 1170, (1997).
>
> DFT nuclear shieldings: T.Helgaker, P.J.Wilson, R.D.Amos, and N.C.Handy. *J.Chem.Phys.*, **113**, 2983, (2000).
>
> CTOCD-DZ nuclear shielding: A.Ligabue, S.P.A.Sauer, P.Lazzeretti. *J.Chem.Phys.*, **118**, 6830, (2003).

The calculation of nuclear shieldings are invoked by the keyword `.SHIELD` in the `**PROPERTIES` input module. Thus a complete input file for the calculation of nuclear shieldings will be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.SHIELD
**END OF DALTON INPUT
```

This will invoke the calculation of nuclear shieldings using London Atomic Orbitals to ensure fast basis set convergence and gauge origin independent results. The natural connection [64] is used in order to get numerically accurate results. By default the center of mass is chosen as gauge origin.

For a SOPPA or SOPPA(CCSD) calculation of nuclear shieldings the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module. For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`. This will also automatically disable the use of London orbitals.

For a CTOCD-DZ calculation of nuclear shieldings the additional keyword `.CTOCD` has to be specified in the `**PROPERTIES` input module. This will automatically disable the use of London orbitals. `.SOPPA` / `.SOPPA` and `.CTOCD` could be used together to get

gauge origin independent SOPPA / SOPPA(CCSD) nuclear shieldings using the CTOCD-DZ formalism. Information about suitable basis set for CTOCD-DZ calculations can be found in the section 10.9.1.

A basis set well suited for the calculation of nuclear shieldings (and indirect nuclear spin–spin coupling constants) is the TZ basis set of Ahlrichs and coworkers [71, 72] with two polarization functions [73]. This basis set is available from the basis set library as TZ2P.

Notice that a general print level of 2 or higher is needed in order to get the individual contributions (relaxation, one- and two-electron expectation values and so on) to the total nuclear shieldings.

If more close control of the different parts of the calculation of the nuclear shieldings is wanted we refer the reader to the section describing the options available. For the calculation of nuclear shieldings, these are the same as listed above for magnetizability calculations.

10.3 Rotational g tensor

> **Reference literature:**
>
> J.Gauss, K.Ruud, and T.Helgaker. *J.Chem.Phys.*, **105**, 2804, (1996).

The calculation of the rotational g tensor is invoked through the keyword `.MOLGFA` in the `**PROPERTIES` input module. A complete input file for the calculation of the molecular g tensor is thus:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.MOLGFA
**END OF DALTON INPUT
```

The molecular g tensor consists of two terms: a nuclear term and a term which may be interpreted as one definition of a paramagnetic part of the magnetizability tensor as described in Ref. [74]. By default the center of mass is chosen as rotational origin, as this corresponds to the point about which the molecule rotates. The use of rotational London atomic orbitals can be turned off through the keyword `.NOLOND`.

For a SOPPA or SOPPA(CCSD) calculation of rotational g tensors the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module. For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE`

FUNCTIONS input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`. This will also disable automatically the use of London orbitals.

The basis set requirements for the rotational g tensors are more or less equivalent with the ones for the molecular magnetizability, that is, the augmented cc-pVDZ of Dunning and Woon [65, 67], available from the basis set library as `aug-cc-pVDZ`.

If more close control of the different parts of the calculation of the rotational g tensor is wanted we refer the reader to the section describing the options available. For the calculation of the molecular g tensor, these are the same as listed above for magnetizability calculations.

10.4 Nuclear spin–rotation constants

Reference literature:

R.Ditchfield. *J.Chem.Phys.*, **56**, 5688 (1972)

J.Gauss, K.Ruud, and T.Helgaker. *J.Chem.Phys.*, **105**, 2804, (1996).

In DALTON the nuclear spin–rotation constants are calculated using rotational orbitals, giving an improved basis set convergence [74]. We use the expression for the spin–rotation constant where the paramagnetic term is evaluated around the center of mass, and thus will only solve three response equations at the most.

An input requesting the calculation of spin–rotation constants will look like

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.SPIN-R
.ISOTOP
    3
    2   1    1
**END OF DALTON INPUT
```

As the nuclear spin–rotation constants depend on the isotopic substitution of the molecule, both through the nuclear magnetic moments and through the center of mass, the isotopic constitution need to be specified if this is different from the most abundant isotopic constitution. Note that some of the most common isotopes do not have a magnetic moment.

For a SOPPA or SOPPA(CCSD) calculation of nuclear spin–rotation constants the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module. For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`. This will also automatically disable the use of London orbitals.

We note that in the current release of DALTON, nuclear spin–rotation constants cannot be calculated employing symmetry-dependent nuclei. Thus for a molecule like N_2, the symmetry plane perpendicular to the molecular bond will have to be removed during the calculation.

10.5 Indirect nuclear spin–spin coupling constants

Reference literature:

O.Vahtras, H.Ågren, P.Jørgensen, H.J.Aa.Jensen, S.B.Padkjær, and T.Helgaker. *J.Chem.Phys.*, **96**, 6120, (1992).

Solvent effects: P.-O.Åstrand, K.V.Mikkelsen, P.Jørgensen, K.Ruud and T.Helgaker. *J.Chem.Phys.*, **108**, 2528, (1998)

SOPPA and SOPPA(CCSD): T.Enevoldsen, J.Oddershede, and S.P.A. Sauer. *Theor. Chem. Acc.*, **100**, 275, (1998)

DFT: T.Helgaker, M.Watson, and N.C.Handy *J.Chem.Phys.*, **113**, 9402, (2000)

As mentioned in the introduction of this chapter, the calculation of indirect nuclear spin–spin coupling constants is a time consuming task due to the large number of contributions to the total spin—spin coupling constant. Still, if all spin—spin couplings in a molecule are wanted, with some restrictions mentioned below, the input will look as follows:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.SPIN-SPIN
**END OF DALTON INPUT
```

This input will calculate the indirect nuclear spin–spin coupling constants between isotopes with non-zero magnetic moments and a natural abundance of more than 1% . This

limit will automatically include proton and ^{13}C spin–spin coupling constants. By default, all contributions to the coupling constants will be calculated.

Often one is interested in only certain kinds of nuclei. For example, one may want to calculate only the proton spin–spin couplings of a molecule. This can be accomplished in two ways: either by changing the abundance threshold so that only this single isotope is included (most useful for proton couplings), or by selecting the particular nuclei of interest.

For a SOPPA or SOPPA(CCSD) calculation of indirect nuclear spin–spin coupling constants the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module. For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`.

All the keywords necessary to control such adjustments to the calculation is given in the section describing the input for the `*SPIN-S` submodule. An input in which we have reduced the abundance threshold as well as selected three atoms will look as:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.SPIN-S
*SPIN-S
.ABUNDA
 0.10
.SELECT
    3
    2    4    5
**END OF DALTON INPUT
```

We refer to the section describing the `*SPIN-S` input module for the complete description of the syntax for these keywords, as well as the numbering of the atoms which are selected.

We also notice that it is often of interest to calculate only specific contributions (usually the Fermi-contact contribution) at a high level of approximation. Sometimes the results obtained with a Hartree–Fock wave function may help in predicting the relative importance of the different contributions, thus helping in the decision of which contributions should be calculated at a correlated level [75]. The calculation of only certain contributions can be accomplished in the input by turning off the different contributions by the keywords

`.NODSO`, `.NOPSO`, `.NOSD`, and `.NOFC`. We refer to the description of the `*SPIN-S` input module for a somewhat more thorough discussion.

If a closer a control of the individual parts of the calculation of indirect nuclear spin–spin coupling constants is wanted, this can be done through the use of keywords in the following input modules:

`*EXPECT` Controls the calculation of one-electron expectation value contribution to the diamagnetic spin–spin coupling constants.

`*GETSGY` Controls the set up of the right-hand sides (gradient terms).

`*LINRES` Controls the solution of the singlet magnetic response equations.

`*TRPRSP` Controls the solution of the triplet magnetic response equations (for Fermi-contact and spin–dipole contributions).

`*RELAX` Controls the multiplication of solution and right hand side vectors into relaxation contributions.

`*SPIN-S` Controls the choice of nuclei for which the spin–spin coupling constants will be calculated, as well as which contributions to the total spin–spin coupling constants are to be calculated.

10.6   Hyperfine Coupling Tensors

Reference literature:

B.Fernandez, P.Jørgensen, J.Byberg, J.Olsen, T.Helgaker, and H.J.Aa.Jensen. *J.Chem.Phys.*, **97**, 3412, (1992).

Solvent effects: B.Fernandez, O.Christiansen, O.Bludsky, P.Jørgensen, K.V. Mikkelsen. *J.Chem.Phys.*, **104**, 629, (1996).

DFT: Z.Rinkevicius, L.Telyatnyk, O.Vahtras and H.Ågren. *J.Chem.Phys.*, **121**, 7614 (2004)

The calculation of hyperfine coupling tensors (in vacuum or in solution) is invoked by the keyword `*ESR` in the `**RESPONSE` input module. Thus a complete input file for the calculation of hyperfine coupling tensors will be:

```
**DALTON INPUT
.RUN RESPONSE
**INTEGRALS
.FC
.SD
**WAVE FUNCTIONS
```

```
.HF
**RESPON
.TRPFLG
*ESR
.ATOMS
 2
 1 2
.FCCALC
.SDCALC
.MAXIT
   30
**END OF DALTON INPUT
```

This will invoke the calculation of hyperfine coupling tensors using the Restricted-Unrestricted methodology [76]. In this approach, the unperturbed molecular system is described with a spin-restricted MCSCF wave function or spin-restricted Kohn–Sham DFT, and when the perturbation—Fermi Contact or Spin Dipole operators—is turned on, the wave function spin relaxes and all first-order molecular properties are evaluated as the sum of the conventional average value term and a relaxation term that includes the response of the wave function to the perturbations.

The selection of a flexible atomic orbital basis set is decisive in these calculations. Dunning's cc-pVTZ or Widmark's basis sets with some functions uncontracted, and one or two sets of diffuse functions and several tight s functions added have been shown to provide accurate hyperfine coupling tensors [77].

If more close control of the different parts of the calculation of hyperfine coupling tensors is wanted, we refer the reader to the sections describing the options available.

10.7   Electronic g-tensors

> **Reference literature:**
>
> ROHF and MCSCF: O.Vahtras, B. Minaev and H.Ågren *Chem.Phys.Lett.*, **281**, 186, (1997).
>
> DFT: Z.Rinkevicius, L.Telyatnyk, P.Sałek, O.Vahtras and H.Ågren. *J.Chem.Phys.*, **119** , 10489, (2003).

The calculation of electronic g-tensors is invoked with the keyword `.G-TENSOR` in the `*ESR` section of the `**RESPONSE` input module

```
**DALTON INPUT
```

```
.RUN RESPONSE
**WAVE FUNCTIONS
.HARTREE-FOCK
**RESPON
*ESR
.G-TENSOR
**END OF DALTON INPUT
```

which gives by default all contributions to the g-tensor to second order in the fine-structure parameter. Keywords following the `.G-TENSOR` keyword will be interpreted as g-tensor options, which are defined in section 26.1.11

10.8 Zero field splitting

> **Reference literature:**
>
> ROHF and MCSCF: O.Vahtras, O.Loboda, B.Minaev, H.Ågren and K.Ruud. *Chem.Phys.*, **279**, 133, (2002).

The calculation of the zero-field splitting is invoked with the keyword `.ZFS` in the `*ESR` section of the `**RESPONSE` input module

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HARTREE-FOCK
**RESPON
*ESR
.ZFS
**END OF DALTON INPUT
```

Note that only the first-order, two-electron, electronic spin–spin contribution is implemented.

10.9 CTOCD-DZ calculations

The DALTON program system can be used to perform calculations of the magnetic properties using the Continuous Transformation of the Origin of the Current Density approach (CTOCD) . Setting the diamagnetic contribution to the current density zero, one obtains fully analytical solutions via equations in closed form for several magnetic properties.

In the present version of DALTON, the following properties can be computed by the CTOCD-DZ approach:

- magnetizability

- nuclear magnetic shielding constant

- shielding polarizability

- hypermagnetizability

the last two properties have to be calculated as quadratic response functions as described in Chapter 12.

10.9.1   General considerations

The CTOCD-DZ approach is competitive with other methods when the dimension of the basis set is not too small, both for magnetizabilities and shieldings [60]. However, for small basis sets the results can be very unreliable. A good basis set for CTOCD-DZ nuclear magnetic shielding calculations, both at the SCF and the correlated level, is the aug-cc-pCVTZ-CTOCD-uc basis set (Ref. [60]), derived from the aug-cc-pVTZ basis set. This basis set is included in the DALTON basis set library.

In SCF calculations, the convergence toward the HF limit is slower than when employing London Orbitals since in the CTOCD-DZ expressions for the magnetic properties (nuclear magnetic shieldings and magnetizabilities) the diamagnetic terms also depend on the first order perturbed density matrix.

On the other hand, the CTOCD scheme does not only fulfill the requirement of translational invariance of the calculated magnetic properties but also guarantees current-charge conservation which is not the case for methods using London Orbitals. In the case of molecules with a vanishing electric dipole moment, CTOCD magnetic susceptibilities are origin independent and the continuity equation is automatically satisfied.

At the present it is possible to obtain CTOCD-DZ magnetizabilities and nuclear magnetic shieldings with SCF, MCSCF, MP2 (using SOPPA) and CCSD (using SOPPA(CCSD)) wave functions via the ABACUS program. Is also possible to compute these properties using the RESPONSE program and using various CC wave functions. Finally is possible to use the quadratic response functions to compute hypermagnetizabilities and shieldings polarizabilities for SCF, MCSCF and CC wave functions.

Using **RESPONSE, one has to be sure that the calculations of the diamagnetic and paramagnetic contributions are both carried out with the gauge origin set at the same positions, since only the full property and not the diamagnetic or paramagnetic contributions are gauge origin independent.

For calculations of nuclear magnetic shieldings (and shielding polarizabilities) using symmetry, the gauge origin has to be placed at the center of mass, otherwise DALTON will give wrong results! This is the default choice of gauge origin. Nuclear magnetic shielding calculations with the gauge origin set at the respective atoms can only be carried without symmetry and by setting the gauge origin on that atom in the **INTEGRAL section.

10.9.2 Input description

> **Reference literature:**
>
> P. Lazzeretti, M. Malagoli and R. Zanasi. *Chem. Phys. Lett.*, **220**, 299 (1994)
>
> P. Lazzeretti *Prog. Nucl. Mag. Res. Spec.*, **220**, 1–88 (2000)
>
> Correlated calculations: A. Ligabue, S. P. A. Sauer and P. Lazzeretti *J.Chem.Phys.*, **118**, 6830, (2003).

The input file for a CTOCD-DZ calculation of the magnetizability and the nuclear magnetic shieldings will be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.CTOCD
.MAGNET
.SHIELD
**END OF DALTON INPUT
```

whereas for the same calculation at SOPPA level it will be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
.MP2
**PROPERTIES
.SOPPA
.CTOCD
.MAGNET
.SHIELD
**END OF DALTON INPUT
```

A coupled cluster calculation of the diamagnetic term of the CTOCD-DZ magnetizability requires the following input:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**INTEGRALS
.RANGMO
.DIPVEL
**WAVE FUNCTIONS
.CC
*CC INPUT
.CCS
.CCSD
*CCLR
.OPERAT
YDIPVEL ZXRANG
ZDIPVEL YXRANG
ZDIPVEL ZXRANG
XDIPVEL YXRANG
XDIPVEL ZXRANG
YDIPVEL YXRANG
YDIPVEL ZYRANG
ZDIPVEL YYRANG
ZDIPVEL ZYRANG
XDIPVEL YYRANG
XDIPVEL ZYRANG
YDIPVEL YYRANG
YDIPVEL ZZRANG
ZDIPVEL YZRANG
ZDIPVEL ZZRANG
XDIPVEL YZRANG
XDIPVEL ZZRANG
YDIPVEL YZRANG
**END OF DALTON INPUT
```

The value for each component of the CTOCD-DZ diamagnetic contribution to the magnetizability can then be obtained as:

$$\chi^{\Delta}_{\alpha\beta} = \epsilon_{\beta\lambda\mu} << \hat{p}_{\lambda}; \hat{r_{O}}_{\mu} \hat{L}_{\alpha} >>$$

where $\hat{r_{O\mu}}\hat{L}_\alpha$ are the $\mu\alpha$RANG operators. Input examples for shieldings and shielding polarizabilities can be found in the test directory.

For **RESPONSE** calculations of these properties the integrals needed in the **INTEGRALS** section are .DIPVEL and .RANGMO for magnetizability, and .DIPVEL and .RPSO for nuclear magnetic shieldings.

Chapter 11

Calculation of optical and Raman properties

This chapter describes the calculation of different optical properties which have been implemented in the DALTON program system. This includes properties related to different kinds of circular dichroism, more specifically vibrational circular dichroism (VCD) as described in Ref. [78], electronic circular dichroism (ECD) as described in Ref. [79], Raman Optical Activity (ROA) as described in Ref. [42], and optical rotation [80, 81].

By default all calculations of optical properties are done with the use of London atomic orbitals in order to enhance the basis set convergence as well as to give the correct physical dependence on the gauge origin.

11.1 Vibrational Circular Dichroism calculations

> **Reference literature:**
>
> K.L.Bak, P.Jørgensen, T.Helgaker, K.Ruud, and H.J.Aa.Jensen. *J.Chem.Phys.*, **98**, 8873, (1993).
>
> K.L.Bak, P.Jørgensen, T.Helgaker, K.Ruud, and H.J.Aa.Jensen. *J.Chem.Phys.*, **100**, 6620, (1994).

The calculation of vibrational circular dichroism is invoked by the keyword `.VCD` in the `**PROPERTIES` input module. Thus a complete input file for the calculation of vibrational circular dichroism will look like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
```

```
**PROPERTIES
.VCD
**END OF DALTON INPUT
```

This will invoke the calculation of vibrational circular dichroism using London atomic orbitals to ensure fast basis set convergence as well as gauge origin independent results. By default the natural connection is used in order to get numerically accurate results [64, 82].

We notice, however, that vibrational circular dichroism only arises in vibrationally chiral molecules. An easy way of introducing vibrational chirality into small molecular systems is by isotopic substitution. This is in DALTON controled in the *VIBANA submodule, and the reader is refered to that section for an exemplification of how this is done.

There has only been a few investigation of basis set requirement for the calculation of VCD given in Ref. [83, 84], and the reader is refered to these references when choosing basis set for the calculations of VCD.

In the current implementation, the .NOCMC option is autmatically turned on in VCD calculations, that is, the coordinate system origin is always used as gauge origin.

We note that if a different force field is wanted in the calculation of the VCD paramaters, this can be obtained by reading in an alternative Hessian matrix with the input

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VCD
*VIBANA
.HESFIL
**END OF DALTON INPUT
```

We note that in the current release of Dalton, Vibrational Circular Dichroism can not be calculated using density function theory, and if requested, the program will stop.

If more close control of the different parts of the calculation of vibrational circular dichroism is wanted, we refer the reader to the sections describing the options available. The input sections that control the calculation of vibrational circular dichroism are:

*AAT Controls the final calculation of the different contributions to the Atomic Axial Tensors.

*GETSGY Controls the set up of both the magnetic and geometric right hand sides (gradient terms).

`*LINRES` Controls the solution of the magnetic response equations.

`*RELAX` Controls the multiplication of solution and right hand side vectors into relaxation contributions.

`*NUCREP` Controls the calculation of the nuclear contribution to the geometric Hessian.

`*TROINV` Controls the use of translation and rotational invariance.

`*ONEINT` Controls the calculation of one-electron contributions to the geometric Hessian.

`*TWOEXP` Controls the calculation of two-electron expectation values to the geometric Hessian.

`*REORT` Controls the calculation of reorthonormalization terms to the geometric Hessian.

`*RESPON` Controls the solution of the geometric response equations.

`*GEOANA` Describes what analysis of the molecular geometry is to be printed.

`*VIBANA` Sets up the vibrational and rotational analysis of the molecule, for instance it's isotopic substitution.

`*DIPCTL` Controls the calculation of the Atomic Polar Tensors (dipole gradient).

11.2 Electronic circular dichroism (ECD) and electronic absorption calculations

> **Reference literature:**
>
> MCSCF: K.L.Bak, Aa.E.Hansen, K.Ruud, T.helgaker, J.Olsen, and P.Jørgensen. *Theor. Chim. Acta.*, **90**, 441, (1995).
>
> DFT: M.Pecul, K.Ruud, and T.Helgaker. *Chem. Phys. Lett.*, **388**, 110, (2004).

The calculation of Electronic Circular Dichroism (ECD) is invoked by the keyword `.ECD` in the ABACUS input module. However, it is also necessary to specify the number of electronic excitations in each symmetry. As ECD only is observed for chiral molecules, such calculations will in general not employ any symmetry, a complete input for a molecule without symmetry will thus look like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
```

```
.HF
**PROPERTIES
.ECD
*EXCITA
.NEXCIT
    3
**END OF DALTON INPUT
```

In this run we will calculate the rotatory strength corresponding to the three lowest electronic excitations (the `.NEXCIT` keyword) using London atomic orbitals. If rotatory strengths obtained without London atomic orbitals is also wanted, this is easily accomplished by adding the keyword `.ROTVEL` in the `*EXCITA` input module.

There are only a few studies of Electronic Circular Dichroism using London atomic orbitals [79, 85], and the results of these investigation indicate that the aug-cc-pVDZ or the d-aug-cc-pVDZ basis set, which is supplied with the DALTON basis set library, is reasonable for such calculations, the double augmentation being important in the case of diffuse/Rydberg-like excited states.

Another property that may often be of interest is the oscillatory strength. This property can be calculated by an input similar to the one for ECD calculation, and for a molecule with C_{2v} symmetry an input would look like:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.EXCITA
*EXCITA
.DIPSTR
.NEXCIT
    3    2    1    0
**END OF DALTON INPUT
```

This input will calculate the dipole strength (`.DIPSTR`) of the 6 lowest electronic excitations distributed in a total of 4 irreducible representations (as in C_{2v}). The dipole strength will be calculated both in length and velocity forms. It is expected that the same requirements for basis set quality applies to this property as for ECD.

For a SOPPA or SOPPA(CCSD) calculation of the oscillator strength the additional keywords `.SOPPA` or `.SOPPA(CCSD)` have to be specified in the `**PROPERTIES` input module.

For SOPPA an MP2 calculation has to be requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module, whereas for SOPPA(CCSD) a CCSD calculation has to be requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module with the `*CC INPUT` option `.SOPPA(CCSD)`.

The two properties may of course be combined in a single run with an input that would then look like (where we also request the rotatory strength to be calculated without the use of London orbitals):

```
**DALTON INPUT
.RUN PROPERITES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.ECD
.EXCITA
*EXCITA
.DIPSTR
.ROTVEL
.NEXCIT
    3
**END OF DALTON INPUT
```

We also note that excitation energies also can be obtained using the RESPONSE program (see Chapter 12). For a more detailed control of the individual parts of the calculation of properties related to electronic excitation energies, we refer to the input modules affecting the different parts of such calculations:

`*EXCITA` Controls the calculation of electronic excitation energies and the evaluation of all terms contributing to for instance dipole strength or electronic circular dichroism.

`*GETSGY` Controls the setup of the necessary right-hand sides.

11.3 Optical Rotation

> **Reference literature:**
>
> MCSCF: T.Helgaker, K.Ruud, K.L.Bak, P.ørgensen, and J.Olsen. *Faraday Discuss.*, **99**, 165, (1994).
>
> P. L. Polavarapu *Mol. Phys.*, **91**, 551, (1997).
>
> DFT: K.Ruud and T.Helgaker. **352**, 553, (2002).

The calculation of optical rotation is a special case of the calculation of Vibrational Raman Optical Activity (see Sec. 11.4), as the tensor determining the optical rotation, the mixed electric-magnetic polarizability, also contributes to vibrational Raman optical activity, although in the latter case it is the geometrical derivatives of the tensor which are the central quantities.

Many of the comments made regarding basis set requirments for VROA calculations will thus also be applicable for the calculation of optical. However, we note that a very extensive basis set investigation of the optical rotation have been presented [86]. A typical input for the calculation of optical rotation would be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
.HF
**PROPERTIES
.OPTROT
*ABALNR
.FREQUENCY
 2
 0.001 0.077318
.THRESH
 1.0D-4
**END OF DALTON INPUT
```

DALTON will always calculate the optical rotation both with and without London atomic orbitals as this has a negligible computational cost compared to the calculation using London atomic orbitals only, since we will anyway have to solve only three response equations corresponding to the perturbing electric field. The optical rotation will only be observed for molecules without symmetry, and by definition the optical rotation will be zero in the static limit. One can approximate the static limit by supplying the program with a very large wavelength (small frequency), as in the example above, in order to be able to compare with approximations that are only valid in the static limit [87, 86].

11.4 Vibrational Raman Optical Activity (VROA)

> **Reference literature:**
>
> MCSCF: T.Helgaker, K.Ruud, K.L.Bak, P.Jørgensen, and J.Olsen. *Faraday Discuss.*, **99**, 165, (1994).
>
> DFT: K.Ruud, T.Helgaker, and P.Bour. *J. Phys. Chem. A*, **106**, 7448, (2002).

The calculation of vibrational Raman intensities and vibrational Raman optical activity (VROA) is one of the more computationally expensive properties that can be evaluated with DALTON.

Due to the time spent in the numerical differentiation, we have chosen to calculate ROA both with and without London atomic orbitals in the same calculation, because the time used in the set-up of the right-hand sides differentiated with respect to the external magnetic field is negligible compared to the time used in the solution of the time-dependent response equations [42]. Because of this, all relevant Raman properties (intensities and depolarization ratios) is also calculated at the same time as ROA.

A very central part in the evaluation of Raman Optical Activity is the evaluation the electric dipole-electric dipole, the electric dipole-magnetic dipole, and the electric dipole-electric quadrupole polarizabilities, and we refer to Section 9.4 for a more detailed description of the input for such calculations.

When calculating Raman intensities and ROA we need to do a numerical differentiation of the electric dipole-electric dipole, the electric dipole-magnetic dipole, and the electric dipole-electric quadrupole polarizabilities along the normal modes of the molecule. The procedure is described in Ref. [42]. We thus need to do a geometry walk of the type numerical differentiation. In each geometry we need to evaluate the electric dipole-electric dipole, the electric dipole-magnetic dipole, and the electric dipole-electric quadrupole polarizabilities. This may be achieved by the following input:

```
**DALTON INPUT
.WALK
*WALK
.NUMERI
**WAVE FUNCTIONS
.HF
*HF INPUT
.THRESH
1.0D-8
**START
```

```
.VROA
*ABALNR
.THRESH
1.0D-7
.FREQUE
     2
0.0 0.09321471
**EACH STEP
.VROA
*ABALNR
.THRESH
1.0D-7
.FREQUE
     2
0.0 0.09321471
**PROPERTIES
.VROA
.VIBANA
*ABALNR
.THRESH
1.0D-7
.FREQUE
     2
0.0 0.09321471
*RESPONSE
.THRESH
1.0D-6
*VIBANA
.PRINT
 2
.ISOTOP
 1 5
 1 1 1 2 3
**END OF DALTON INPUT
```

This is the complete input for a calculation of VROA on the CFHDT molecule. In addition to the keyword `.VROA` in the different ABACUS input modules, we still need to tell the program that frequencies of the laser field are to be read in the `*ABALNR` section.

The only isotopic substitution of this molecule that shows vibrational optical activity

is the one containing one hydrogen, one deuterium and one tritium nucleus. If we want the center-of-mass to be the gauge origin for the VROA calculation not employing London atomic orbitals, this have to be reflected in the specification of the isotopic constitution of the molecule, see Chapter 23. We note that a user specified gauge origin can be supplied with the keyword `.GAUGEO` in the ABACUS input modules. The gauge origin can also be chosen as the origin of the Cartesian Coordinate system (0,0,0) by using the keyword `.NOCMC`. Note that neither of these options will affect the results obtained with London orbitals.

The input in the `*ABALNR` input section should be self-explanatory from the discussion of the frequency dependent polarizability in Sec. 9.4. Note that because of the numerical differentiation the response equations need to be converged rather tightly ($1.0 \cdot 10^{-7}$). Remember also that this will require you to converge your wave function more tightly than is the default.

The numerical differentiation is invoked through the keyword `.NUMERI` in the `*WALK` submodule. Note that this will automatically turn off the calculation of the molecular Hessian, putting limitations on what properties may be calculated during a ROA calculation. Because of this there will not be any prediction of the energy at the new point.

It should also be noted that the program in a numerical differentiation will step plus and minus one displacement along each Cartesian coordinate of all nuclei, as well as calculating the property in the reference geometry. Thus, for a molecule with N atoms the properties will be calculated in a total of $2 * 3 * N + 1$ points, which for a 5 atom molecule will amount to 31 points. The default maximum number of steps of the program is 20. By default the program will for numerical differentiation calculations reset the maximum number of iterations to $6N+1$. However, it is also possible to set the number of iterations explicitly in the general input module using the keyword `.MAX IT` described in Section 21.1.

The default step length in the numerical integration is 10^{-4} a.u., and this step length may be adjusted by the keyword `.DISPLA` in the `*WALK` module. The steps are taken in the Cartesian directions and not along normal modes. This enables us to study a large number of isotopically substituted molecules at once, as the London orbital results for ROA does not depend on the choice of gauge origin. This is done in the `**PROPERTIES` input module, but as only one isotopic substituted species show optical activity, we have only requested a vibrational analysis for this species.

We note that as in the case of Vibrational Circular Dichroism, a different force field may be used in the estimation of the VROA intensity parameters. Indeed, a number of force fields can be used to estimate the VROA parameters obtained with a given basis set through the input:

```
**DALTON INPUT
.WALK
.ITERATION
```

```
 31
*WALK
.NUMERI
**PROPERTIES
.VROA
.VIBANA
*ABALNR
.THRESH
1.0D-7
.FREQUE
      2
0.0 0.09321471
*RESPONSE
.THRESH
1.0D-6
*VIBANA
.HESFIL
.PRINT
 2
.ISOTOP
 1 5
 1 1 1 2 3
**END OF DALTON INPUT
```

by copying different `DALTON.HES` files to the scratch directory, which in turn is read through the keyword `.HESFIL`. By choosing the start iteration to be 31 through the keyword `.ITERAT`, we tell the program that the walk has finished (for CHFDT with 31 points that need to be calculated). However, this requires that all information is available in the `DALTON.WLK` file.

Concerning basis sets requirement for Raman Optical Activity, a thorough investigation of the basis set requirements for the circular intensity differences (CIDs) in VROA was presented by Zuber and Hug [88]. They also presented a close-to-minimal basis set that yields high-quality CIDs. The force fields do, however, have to be determined using larger basis (aug-cc-pVTZ) and including electron correlation for a reliable prediction of VROA spectra.

Chapter 12

Getting the property you want

In the preceding chapters we have shown how to calculate a number of properties that are associated with specific spectroscopic applications. For HF, DFT, SOPPA, and MCSCF these properties are in part calculated in the RESPONSE program, but given that a large number of standard calculations usually are carried out in a similar fashion, some applications have a simplified input (under **PROPERTIES), and an appealing output that meets common demands (*e.g.* customary unit conversions). For CC calculations of properties, see Chapter 27. In this chapter we describe how to set up the input for calculating a general property that can be defined in terms of electronic response functions.

12.1   General considerations

> **Reference literature:**
>
> Response theory: Jeppe Olsen and Poul Jørgensen, *J. Chem. Phys.* **82**, 3235, (1985)

A response function is a measure of how a property of a system changes in the presence of one or more perturbations. With our notation (see *e.g.* Ref. [48]), $\langle\langle A; B \rangle\rangle_{\omega_b}$, $\langle\langle A; B, C \rangle\rangle_{\omega_b,\omega_c}$, and $\langle\langle A; B, C, D \rangle\rangle_{\omega_b,\omega_c,\omega_d}$ denote linear, quadratic and cubic response functions, respectively, which provide the first, second, and third-order corrections to the expectation-value of A, due to the perturbations B, C, and D, each of which is associated with a frequency ω_b, ω_c, and ω_d. Often the perturbations are considered to be external monochromatic fields, or static (*e.g.* relativistic) perturbations, in which case the frequency is zero. In general, the perturbations B, C, and D represent Fourier components of an arbitrary time-dependent perturbation.

12.2 Input description

In this section we describe a few minimal input examples for calculating some molecular properties that can be expressed in terms of linear, quadratic, and cubic response functions. Note that only one of these three different orders of response functions can be requested in the same calculation.

For more information on keywords, see the Reference Manual, chapter 26.

12.2.1 Linear response

> **Reference literature:**
>
> Singlet linear response: Poul Jørgensen, Hans Jørgen Aagaard Jensen, and Jeppe Olsen, *J. Chem. Phys.* **89**, 3654, (1988)
>
> Triplet linear response: Jeppe Olsen, Danny L. Yeager, and Poul Jørgensen, *J. Chem. Phys.* **91**, 381, (1989)
>
> SOPPA linear response: Martin J. Packer, Erik K. Dalskov, Thomas Enevoldsen, Hans Jørgen Aagaard Jensen and Jens Oddershede, *J. Chem. Phys.*, **105**, 5886, (1996)
>
> SOPPA(CCSD) linear response: Stephan P. A. Sauer, *J. Phys. B: At. Opt. Mol. Phys.*, **30**, 3773, (1997)
>
> DFT open-shell linear response: Zilvinas Rinkevicius, Ingvar Tunell, Paweł Sałek, Olav Vahtras, and Hans Ågren, *J. Chem. Phys.*, **119**, 34, (2003)

A well-known example of a linear response function is the polarizability. A typical input for SCF static and dynamic polarizability tensors $\alpha_{ij}(-\omega;\omega) \equiv -\langle\langle x_i; x_j \rangle\rangle_\omega$ for a few selected frequencies (in atomic units) will be:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
**RESPONSE
*LINEAR
.DIPLEN
.FREQUENCIES
 3
 0.0 0.5 1.0
**END OF DALTON INPUT
```

The `.DIPLEN` keyword has the effect of defining the A and B operators as all components of the electric dipole operator.

A Second Order Polarization Propagator Approximation (SOPPA)[49, 50, 51] calculation of linear response functions can be invoked if the additonal keyword `.SOPPA` is specified in the `**RESPONSE` input module and an MP2 calculation is requested by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module. A typical input for SOPPA dynamic polarizability tensors will be:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
.MP2
**RESPONSE
.SOPPA
.NOITRA
*LINEAR
.DIPLEN
.FREQUENCIES
 3
 0.0 0.5 1.0
**END OF DALTON INPUT
```

The `.NOITRA` keyword has the effect that the transformation of the two electron integrals necessary for a MP2 and SOPPA calculation is only perfomed once in the `**WAVE FUNCTIONS` module.

A Second Order Polarization Propagator Approximation with Coupled Cluster Singles and Doubles Amplitudes - SOPPA(CCSD)[53] calculation of linear response functions can be invoked if the additonal keyword `.SOPPA(CCSD)` is specified in the `**RESPONSE` input module and an CCSD calculation is requested by the keyword `.CC` in the `**WAVE FUNCTIONS` input module. A typical input for SOPPA(CCSD) dynamic polarizability tensors will be:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
.CC
*CC INPUT
.SOPPA(CCSD)
**RESPONSE
```

```
.SOPPA(CCSD)
*LINEAR
.DIPLEN
.FREQUENCIES
 3
 0.0 0.5 1.0
**END OF DALTON INPUT
```

The linear response function contains a wealth of information about the spectrum of a given Hamiltonian. It has poles at the excitation energies, relative to the reference state (not necessarily the ground state) and the corresponding residues are transition moments between the reference and excited states. To calculate the excitation energies and dipole transition moments for the three lowest excited states in the fourth symmetry, a small modification of the input above will suffice;

```
**RESPONSE
*LINEAR
.SINGLE RESIDUE
.DIPLEN
.ROOTS
 0 0 0 3
```

12.2.2   Quadratic response

> **Reference literature:**
>
> Singlet quadratic response: Hinne Hettema, Hans Jørgen Aa. Jensen, Poul Jørgensen, and Jeppe Olsen, *J. Chem. Phys.* **97**, 1174, (1992)
>
> Triplet quadratic response: Olav Vahtras, Hans Ågren, Poul Jørgensen, Hans Jørgen Aa. Jensen, Trygve Helgaker, and Jeppe Olsen, *J. Chem. Phys.* **97**, 9178, (1992)
>
> Integral direct quadratic response: Hans Ågren, Olav Vahtras, Henrik Koch, Poul Jørgensen, and Trygve Helgaker, *J. Chem. Phys.* **98**, 6417, (1993)
>
> DFT singlet quadratic response: Paweł Sałek, Olav Vahtras, Trygve Helgaker, and Hans Ågren, *J. Chem. Phys.* **117**, 9630, (2002)
>
> DFT triplet quadratic response: Ingvar Tunell, Zilvinas Rinkevicius, Olav Vahtras, Paweł Sałek, Trygve Helgaker, and Hans Ågren, *J. Chem. Phys.* **119**, 11024, (2003)

An example of a quadratic response function is the first hyperpolarizability. If we are interested in $\beta_{zzz} \equiv -\langle\langle z; z, z\rangle\rangle_{0,0}$ only, we may use the following input:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
**RESPONSE
*QUADRATIC
.DIPLNZ
**END OF DALTON INPUT
```

When no frequencies are given in the input, the static value is assumed by default. If we wish to calculate dynamic hyperpolarizabilities we supply frequencies, but in this case we have two frequencies ω_b, ω_c which are given by the keywords `.BFREQ` and `.CFREQ` (see the Reference Manual, chapter 26). The non-zero linear response functions from the operators can be generated with no additional computational costs, and all $\langle\langle A; B \rangle\rangle_{\omega_b}$ results will also be printed (in this example α_{zz}).

The residue of a quadratic response function gives two-photon transition amplitudes. For such a calculation we supply the same extra keywords as in the linear case (Sec. 12.2.1):

```
**RESPONSE
*QUADRATIC
.DIPLNZ
.SINGLE RESIDUE
.ROOTS
 2 0 0 0
```

which in this case means the two-photon transition amplitude between the reference state and the first two excited states in the first symmetry. In general the residue of a quadratic response function corresponds to the induced transition moment of an operator A due to a perturbation B. The C operator is arbitrary and is not specified. A typical example is the dipole matrix element between a singlet and triplet state that is induced by spin-orbit coupling (phosphorescence). For this special case we have the keyword, `.PHOSPHORESCENCE` under `*QUADRATIC`, which sets A to electric dipole operators and B to spin-orbit operators.

The residue of a quadratic response function can be used to identify the two-photon transition amplitudes. The input below refers to the calculation of the two-photon absorption from the ground state to the first 3 excited states in point group symmetry one. In the program output the two-photon transition matrix element is given as well as the two-photon transition probability relevant for an isotropic gas or liquid. The evaluation of the transition probabilities can be done based on the transition matrix elements although they, in principle, are connected with the imaginary part of the second hyperpolarizability. The absorption cross sections are evaluated assuming a monochromatic light source that is either linearly or circularly polarized.

```
**RESPONSE
*QUADRATIC
.TWO-PHOTON
.ROOTS
 3 0 0 0
```

Another special case of a residue of the quadratic response function is the $\mathcal{B}(0 \to f)$ term of magnetic circular dichroism (MCD).

```
**RESPONSE
*QUADRATIC
.SINGLE RESIDUE
.ROOTS
 2 2 0 0
.MCDBTERM
```

For each dipole-allowed excited state among those specified in `.ROOTS`, the `.MCDBTERM` keyword automatically calculates all symmetry allowed products of the single residue of the quadratic response function for A corresponding to the electric dipole operator and B to the angular momentum operator with the single residue of the linear response function for C equal to the electric dipole operator. In other words, the mixed electric dipole—magnetic dipole two-photon transition moment for final state f times the dipole one-photon moment for the same state f. Note that in the current implementation (for SCF and MCSCF), degeneracies between excited states may lead to numerical divergencies. The final $\mathcal{B}(0 \to f)$ must be obtained from a combination of the individual components, see the original paper [17].

It is possible to construct double residues of the quadratic response function, the interpretation of which is transition moments between two excited states. Specifying `.DOUBLE` in the example above thus gives the matrix elements of the z-component of the dipole moment between all excited states specified in `.ROOTS`. Note that the diagonal contributions gives not the expectation value in the excited state, but rather the difference relative to the reference state expectation value.

12.2.3 Cubic response

> **Reference literature:**
>
> SCF cubic response: Patrick Norman, Dan Jonsson, Olav Vahtras, and Hans Ågren, *Chem. Phys. Lett.* **242**, 7, (1995)
>
> MCSCF cubic response: Dan Jonsson, Patrick Norman, and Hans Ågren, *J. Chem. Phys.* **105**, 6401, (1996)

All components of the static second hyperpolarizability defined as $\gamma_{ijkl}(-0;0,0,0) \equiv -\langle\langle x_i; x_j, x_k, x_l \rangle\rangle_{000}$, may be obtained by the following input

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
**RESPONSE
*CUBIC
.DIPLEN
**END OF DALTON INPUT
```

As mentioned above (Sec. 12.2.2), the "diagonal" double residue of the quadratic response function is the change in the expectation value relative to the reference state. The analogue for cubic response functions is the change in polarizability relative to the reference state polarizability, which is demonstrated by the following input.

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
**RESPONSE
*CUBIC
.DOUBLE RESIDUE
.DIPLNZ
.FREQUENCIES
 1
 0.01
.ROOTS
  1 0 0 0
**END OF DALTON INPUT
```

Chapter 13

Direct and parallel calculations

In this chapter we briefly discuss aspects connected to direct and parallel methods as implemented in the DALTON program.

13.1 Direct methods

Reference literature:

H. Ågren, O. Vahtras, H. Koch, P. Jørgensen, and T. Helgaker. *J. Chem. Phys.*, **98**, 6471, (1993).

K. Ruud, D. Jonsson, P. Norman, H. Ågren, T. Saue, H. J. Aa. Jensen, P. Dahle, and P. Dahle. *J. Chem. Phys.* **108**, 7973 (1998).

The entire SCF and Kohn–Sham DFT part of the DALTON code is direct, including all derivative two-electron integrals, and all the way up to the cubic response function. To perform a direct calculation, all that is required is to add the keyword `.DIRECT` in the general input section, as indicated in the following input example for the calculation of nuclear shieldings in a direct fashion:

```
**DALTON INPUT
.RUN PROPERTIES
.DIRECT
**WAVE FUNCTIONS
.HF
**PROPERTIES
.SHIELD
**END OF DALTON INPUT
```

By default the two-electron integrals will be screened [89], using an increasingly tighter integral screening threshold during the SCF iterations. Thus, direct calculations

calculated using integral screening will, when converged, be almost as exact as the results obtained with integral screening turned off. The thresholds can be changed with the keywords `.IFTHRS` and `.ICEDIF` in the `*TWOINT` section of the `**INTEGRALS` input module. To turn off integral screening in direct or parallel calculations altogether, `.IFTHRS` should be set to 20 or larger.

13.2   Parallel methods

> **Reference literature:**
>
> P. Norman, D. Jonsson, H. Ågren, P. Dahle, K. Ruud, T. Helgaker, and H. Koch. *Chem. Phys. Lett.*, **253**, 1, (1996).

As for direct methods, the entire Hartree–Fock and Kohn–Sham DFT parts of the DALTON program has been parallelized using MPI as message passing interface. The use of the parallel code requires, however, that the code has been installed as a parallel code, which is being determined during the building of the program as described in Section 3.3.

All that is needed to do of changes in the `DALTON.INP` file is to add the keyword `.PARALLEL` in the general input section, as demonstrated for a calculation of vibrational frequencies:

```
**DALTON INPUT
.RUN PROPERTIES
.PARALLEL
**WAVE FUNCTIONS
.HF
**PROPERTIES
.VIBANA
**END OF DALTON INPUT
```

The number of nodes to be used in the calculation is requested to the `dalton` run script after the `-N` option (see Section 5.4), or as stated in local documentation. Note that the master/slave paradigm employed by DALTON will leave the master mainly doing sequential parts of the calculation and distribution of tasks, thus very little computation compared to the `N-1` slaves, see Ref. [90].

By default the two-electron integrals will be screened [89], using an increasingly tighter integral screening threshold during the SCF iterations. Thus, direct calculations calculated using integral screening will, when converged, be almost as exact as the results obtained with integral screening turned off. The thresholds can be changed with the keywords `.IFTHRS` and `.ICEDIF`. To turn of integral screening in direct or parallel calculations altogether, `.IFTHRS` should be set to 20 or larger.

Chapter 14

Finite field calculations

Despite the large number of properties that in principle can be calculated with DALTON, it is often of interest to study the dependence of these properties under an external electric perturbation. This can be easily achieved by adding static electric fields, and thus increase the number of properties that can be calculated with DALTON.

In the next section we comment briefly on some important aspects of finite electric field calculations, and the following section describes the input for finite field calculations.

14.1   General considerations

The presence of an external electric field can be modeled by adding a term to our ordinary field-free, non-relativistic Hamiltonian corresponding to the interaction between the dipole moment operator and the external electric field:

$$\mathcal{H} = \mathcal{H}^0 - \mathbf{E}\mathbf{d}_e \tag{14.1}$$

where $\mathbf{d}_e$ is the electric dipole moment operator defined as

$$\mathbf{d}_e = \sum_i \mathbf{r}_i \tag{14.2}$$

and $\mathcal{H}^0$ is our ordinary field-free, non-relativistic Hamiltonian operator. It is noteworthy that we do not include the nuclear dipole moment operator, and the total electronic energy will thus depend on the position of the molecule in the Cartesian coordinate frame.

The electric field dependence of different molecular properties are obtainable by adding fields in different directions and with different signs and then extract the information by numerical differentiation. Note that care has to be taken to choose a field that is weak enough for the numeric differentiation to be valid, yet large enough to give numerically significant changes in the molecular properties (see for instance Ref. [91]). Note also that

it may be necessary to increase the convergence threshold for the solution of the response equations if molecular properties are being evaluated.

Whereas the finite field approach may be combined with any property that can be calculated with the RESPONSE program, more care need to be taken if the finite field method is used with the ABACUS program. Properties that involve perturbation-dependent basis sets, like nuclear shieldings and molecular Hessians, will often introduce extra reorthonormalization terms due to the finite field operator, and care has to be taken to ensure that these terms indeed have been included in DALTON.

NOTE: In the current release, the only properties calculated with perturbation dependent basis sets that may be numerically differentiated using finite field, are the nuclear shieldings and magnetizabilities using the implementation described in Ref. [92], and molecular gradients.

14.2 Input description

> **Reference literature:**
>
> Shielding and magnetizability polarizabilities: A.Rizzo, T.Helgaker, K.Ruud, A.Barszczewicz, M.Jaszuński and P.Jørgensen. *J.Chem.Phys.*, **102**, 8953, (1995).

The necessary input for a finite-field calculation is given in the **INTEGRALS and **WAVE FUNCTIONS input modules. A typical input file for an finite field SCF calculation of the magnetizability of a molecule will be:

```
**DALTON INPUT
.RUN PROPERTIES
**INTEGRALS
.DIPLEN
**WAVE FUNCTIONS
.HF
*HAMILTONIAN
.FIELD
 0.003
 XDIPLEN
**PROPERTIES
.MAGNET
**END OF DALTON INPUT
```

In the **INTEGRALS input module we request the evaluation of dipole length integrals, as these correspond to the electric dipole operator, and will be used in SIRIUS for

evaluating the interactions between the electric dipole and the external electric field. This is achieved in the *HAMILTONIAN input module, where the presence of an external electric field is signaled by the keyword .FIELD. On the next line, the strength of the electric field (in atomic units) is given, and on the following line we give the direction of the applied electric field (XDIPLEN, YDIPLEN, or ZDIPLEN). Several fields may of course be applied at the same time. In comparison with DALTON 1.2, the present version of DALTON can also calculate the nuclear shielding polarizabilities with respect to an external electric field gradient using London atomic orbitals, both using the traceless quadrupole operator .THETA and the second moment of charge operator .SECMOM.

Chapter 15

Solvent calculations

This chapter describes the Multiconfigurational Self-Consistent Reaction Field (MCSCRF) model as implemented in DALTON. The first section describes some considerations about the implementation and the range of properties that may be evaluated with the present MCSCRF implementation. The second section gives two input examples for MCSCRF calculations.

15.1 General considerations

DALTON has the possibility of modelling the effect of a surrounding linear, homogeneous dielectric medium on a variety of molecular properties using SCF or MCSCF wave functions. This is achieved by the Multiconfigurational Self-Consistent Reaction Field (MCSCRF) approach [93, 94], where the solute is placed in a spherical cavity and surrounded by the dielectric medium. The solvent response to the presence of the solute is modelled by a multipole expansion, in DALTON in principle to infinite order, but practical applications show that the multipole expansion is usually converged at order $L = 6$.

In DALTON the solvent model is implemented both for SCF, DFT and MCSCF wave functions in a self-consistent manner as describes in Ref. [93, 94]. In MCSCF calculations where MP2 orbitals is requested as starting orbitals for the MCSCF optimization, the solvent model will not be added before entering the MCSCF optimization stage, so MP2 gas-phase orbitals can be used as starting guess even though the solvent model has not been implemented for this wave function model. Note also that differential densities will be disabled in direct calculations when the solvent model is employed.

As regards molecular properties, the solvent model has so far been extended to singlet linear, quadratic and cubic response, and triplet linear response in the RESPONSE program, both using equilibrium and non-equilibrium solvation. A number of properties and excitation energies can be calculated with the (MC)SCRF model, and several studies of

such properties have been presented, and we refer to these papers for an overview of what can currently be calculated with the approach [95, 96], including ESR hyperfine coupling constants [97].

In addition, a non-equilibrium solvation model has been implemented for molecular energies [98]. This model in needed when studying processes where the charge distribution of the solute cannot be expected to be in equilibrium with the charge distribution of the solvent, *e.g.* when comparing with experiments where light has been used as a perturbation.

In the ABACUS program, the solvent model has been implemented for geometric distortions and nuclear shieldings and magnetizabilities, and of course all the properties that do not use perturbation-dependent basis sets, such as for instance indirect spin-spin coupling constants. This is noteworthy, as although the program will probably give results for most results calculated using the solvent model, these results will not necessarily be theoretically correct, due to lack of reorthonormalization contributions that have not been considered in the program. We therefore give a fairly complete literature reference of works that have been done with the program [99, 100]. Properties not included in this list are thus not trustworthy with the current version of DALTON.

15.2 Input description

Reference literature:

General reference: K.V.Mikkelsen, E.Dalgaard, P.Svanstrøm. *J.Phys.Chem*, **91**, 3081, (1987).

General reference: K.V.Mikkelsen, H.Ågren, H.J.Aa.Jensen, and T.Helgaker. *J.Chem.Phys.*, **89**, 3086, (1988).

Non-equilibrium solvation: K.V.Mikkelsen, A.Cesar, H.Ågren, H.J.Aa.Jensen. *J.Chem.Phys.*, **103**, 9010, (1995).

Linear singlet response: K.V.Mikkelsen, P.Jørgensen, H.J.Aa.Jensen. *J.Chem.Phys.*, **100**, 6597, (1994).

Linear triplet response: P.-O.Åstrand, K.V.Mikkelsen, P.Jørgensen, K.Ruud and T.Helgaker. *J. Chem. Phys.*, **108**, 2528 (1998).

Hyperfine couplings: B.Fernandez, O.Christensen, O.Bludsky, P.Jørgensen, K.V.Mikkelsen. *J.Chem.Phys.*, **104**, 629, (1996).

Magnetizabilities and nuclear shieldings: K.V.Mikkelsen, P.Jørgensen, K.Ruud, and T.Helgaker. *J.Chem.Phys.*, **107**, 1170, (1997).

Molecular Hessian: P.-O.Åstrand, K.V.Mikkelsen, K.Ruud and T.Helgaker. *J.Phys.Chem.*, **100**, 19771, (1996).

Spin-spin couplings: P.-O.Åstrand, K.V.Mikkelsen, P.Jørgensen, K.Ruud and T.Helgaker. *J. Chem. Phys.*, **108**, 2528 (1998).

The necessary input for a solvent calculation is given in the `**INTEGRALS` and `**WAVE FUNCTIONS` input modules. A typical input file for an SCF calculation of the nuclear shielding constants of a molecule in a dielectric medium will look like:

```
**DALTON INPUT
.RUN PROPERTIES
**INTEGRALS
*ONEINT
.SOLVENT
 10
**WAVE FUNCTIONS
.HF
*SOLVENT
.DIELECTRIC CONSTANT
 78.5
.MAX L
 10
.CAVITY
 3.98
**PROPERTIES
.SHIELD
**END OF DALTON INPUT
```

In `**INTEGRALS` we request the evaluation of the undifferentiated solvent multipole integrals as given in for instance Ref. [94] by the keyword `.MAX L` in the `*ONEINT` submodule. We request all integrals up to $L = 10$ to be evaluated. This is needed if static or dynamic (response) properties calculations are to be done, but is not needed for a run of the wave function only (`**WAVE FUNCTIONS`).

In `**WAVE FUNCTIONS` there is a separate input module for the solvent input, headed by the name `*SOLVENT`. We refer to Sec. 24.2.16 for a presentation of all possible keywords in this submodule. The interaction between the solute and the dielectric medium is characterized by three parameters; the dielectric constant, the cavity size, and the order of the multipole expansion. In the above input we have requested a dielectric constant of 78.5 (corresponding to water) through the keyword `.DIELECTRIC CONSTANT`, a cavity radius of 3.98 atomic units with the keyword `.CAVITY`, and the multipole expansion is to include all terms up to $L = 10$, as can be seen from the keyword `.MAX L`. Note that this number cannot be larger than the number given for `.SOLVENT` in the `*ONEINT` input module.

15.2.1   Geometry optimization

In the present release of the DALTON program, there are certain limitations imposed on the optimizing geometries using the solvent model. Only second-order geometry optimizations are available, and only through the general `*WALK` module. Thus the input for an SCF geometry optimization with the solvent model would look like:

```
**DALTON INPUT
.WALK
**INTEGRALS
*ONEINT
.SOLVENT
 10
**WAVE FUNCTIONS
.HF
*SOLVENT
.DIELECTRIC CONSTANT
 78.5
.MAX L
 10
.CAVITY
 3.98
**PROPERTIES
.VIBANA
.SHIELD
**END OF DALTON INPUT
```

15.2.2   Non-equilibrium solvation

This example describes calculations for non-equilibrium solvation. Usually one starts with a calculation of a reference state (most often the ground state) with equilibrium solvation, using keyword `.INERSFINAL`. The interface file is then used (without user interference) for a non-equilibrium excited state calculation; keyword `.INERSINITIAL`.

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**INTEGRALS
*ONEINT
.SOLVENT
 10
```

```
**WAVE FUNCTIONS
.TITLE
 2-RAS(2p2p') : on F+ (1^D) in Glycol
 Widmark (5432)-ANO Basis set
.MCSCF
*CONFIGURATION INPUT
.SPIN MULTIPLICITY
 1
.SYMMETRY
 1
.INACTIVE ORBITALS
 1  0  0  0  0  0  0  0
.ELECTRONS
 6
.RAS1 SPACE
 0  0  0  0  0  0  0  0
.RAS2 SPACE
 1  2  2  0  2  0  0  0
.RAS3 SPACE
 8  4  4  3  4  3  3  1
.RAS1 ELECTRONS
 0  0
.RAS3 ELECTRONS
 0  2
*OPTIMIZATION
.NEO ALWAYS
.OPTIMAL ORBITAL TRIAL VECTORS
.MAX CI
 30
*ORBITAL INPUT
.MOSTART       | Note, we assume the existence of an SIRIUS.RST file
 NEWORB
*CI VECTOR
.STARTOLDCI    | Note, we assume the existence of an SIRIUS.RST file
*SOLVENT
.CAVITY
 2.5133D0
.INERSINITIAL     | initial state inertial polarization
```

```
 37.7D0  2.050D0  | static and optic dielectric constants for Glycol
.MAX L
 10
**END OF DALTON INPUT
```

Chapter 16

Vibrational corrections

DALTON provides an efficient automated procedure for calculating rovibrationally averaged molecular r_α geometries, as well as an automated procedure for calculating vibrational averages of a large range of second-order molecular properties, for SCF and MCSCF wave functions. In the current implementation, it is not possible to exploit point-group symmetry, and one must ensure that the symmetry is turned off in the calculation.

Reference literature:

Effective geometries: P.-O. Åstrand, K. Ruud and P. R. Taylor. *J.Chem.Phys*, **112**, , 2655 (2000).

Vibrational averaged properties: K. Ruud, P.-O. Åstrand and P. R. Taylor. *J.Chem.Phys.*, **112**, 2668, (2000).

Temperature and isotope effects: K. Ruud, J. Lounila and J. Vaara. *J.Chem.Phys.*, to be published.

16.1 Effective geometries

The (ro)vibrationally averaged geometries can be calculated from a knowledge of part of the cubic force field

$$\langle r_i \rangle = r_{e,i} - \frac{1}{4\omega_i^2} \sum_{j=1}^{3N-6} \frac{V_{ijj}^{(3)}}{\omega_j} \tag{16.1}$$

where the summation runs over all normal modes in the molecule and where ω_i is the harmonic frequency of normal mode i and $V_{ijj}^{(3)}$ is the cubic force field. A typical input for determining (ro)vibrationally averaged Hartree–Fock geometries for different water isotopomers will look like

```
**DALTON INPUT
```

```
.WALK
*WALK
.ANHARM
.DISPLACEMENT
0.001
.TEMPERATURES
 4
 0.0 300.0 500.0 1000.0
**WAVE FUNCTIONS
.HF
*SCF INPUT
.THRESH
 1.0D-10
**START
*RESPONS
.THRESH
 1.0D-5
**EACH STEP
*RESPONS
.THRESH
 1.0D-5
**PROPERTIES
.VIBANA
*RESPONS
.THRESH
 1.0D-5
*VIBANA
.ISOTOP
 3 3
 1 2 1
 1 2 2
 2 1 1
**END OF DALTON INPUT
```

The calculation of (ro)vibrationally averaged geometries are invoked be the keyword .ANHARM in the *WALK input module. In this example, the full cubic force field will be determined as first derivatives of analytical molecular Hessians. This will be done in Cartesian coordinates, and the calculation will therefore require the evaluation of $6K + 1$ analytical Hessians, where K is the number of atoms in the molecules. Although expensive, it allows

(ro)vibrational corrections to be calculated for any isotopic species, in the above example for H_2 ^{16}O, HD ^{16}O, D_2 ^{16}O, H_2 ^{18}O. This is directed by the keyword `.ISOTOP`. We note that the most abundant isotope will always be calculated, and is therefore not included in the list above.

We have requested that rovibrationally averaged geometries be calculated for 5 different temperatures. By default, these geometries will include centrifugal distortions [16]. This can be turned by using the keyword `.NO CENT` in the `*WALK` input module.

By default, the numerical differentiation will use a step length of 0.0001 bohr. Experience show this to be too short [14], and we have therefore changed this to be 0.001 bohr in the example above by the use of the keyword `.DISPLACMENT` in the `*WALK` input module.

If only one (or a few) isotopic species are of interest, we can significantly speed up the calculation of the (ro)vibrationally averaged geometries by doing the numerical differentiation in the normal coordinates of the isotopic species of interest. This can be requested through the keyword `.NORMAL`. The relevant part of the cubic force field is then calculated as numerical second derivatives of analytical gradients. We note that the suggested step length in this case should be set to 0.0075 [14]. We note that we will still need to calculate one analytical Hessian in order to determine the normal coordinates.

The default maximum number of iterations is 20. However, DALTON will automatically reset the maximum number of iterations to $6K+1$ in case of vibrational averaging calculations. The maximum number of iterations can also be set explicitly by using the keyword `.MAX IT` in the `**DALTON INPUT` module.

16.2   Vibrational averaged properties

The change in the geometry accounts for part of the contribution to a vibrationally averaged property, namely that due to the anharmonicity of the potential [101]. Although this term is important, we need to include also the contribution from the averaging of the molecular property over the harmonic oscillator wave function in order to get an accurate estimate of the vibrational corrections to the molecular property.

At the effective geometry, this contribution to for instance the nuclear shielding constants can be obtained from the following input

```
**DALTON INPUT
.WALK
*WALK
.VIBAVE
.DISPLACEMENT
0.05
.TEMPERATURES
```

```
 1
 300.0
**WAVE FUNCTIONS
.HF
*SCF INPUT
.THRESH
 1.0D-10
**START
.SHIELD
*LINRES
.THRESH
 1.0D-6
*RESPONS
.THRESH
 1.0D-5
**EACH STEP
.SHIELD
*LINRES
.THRESH
 1.0D-6
*RESPONS
.THRESH
 1.0D-5
**END OF DALTON INPUT
```

This input will calculate the harmonic contribution to the (ro)vibrational average to the nuclear shielding constants at 300K for ^{17}ODH. It is important to realize that since each isotopic species for each temperature will have its own unique (ro)vibrationally averaged geometry, we will have to calculate the harmonic contribution for each temperature and each isotopic species separately. The isotopic constitution is specified in the MOLECULE.INP file as described in Chapter 23.

We note that we may reuse the property derivatives from a different geometry for calculating the harmonic contribution to the vibrational correction at the given geometry by using the keyword .REUSE in the *WALK module. A new force field is calculated, but the property derivatives are assumed to remain unchanged. The approximation has been tested and been shown to account, through the change in the effective geometry for different temperatures, for a very large fraction of the temperature effects on molecular properties [16].

This calculation will always be done in normal coordinates, and the recommended step length is 0.05 [15]. As for the calculation of (ro)vibrationally averaged geometries in

normal coordinates, the calculation requires the determination of one analytical Hessian in order to determine the harmonic force field.

The default maximum number of iterations is 20. However, DALTON will automatically reset the maximum number of iterations to $6K+1$ in case of vibrational averaging calculations. The maximum number of iterations can also be set explicitly by using the keyword `.MAX IT` in the `**DALTON INPUT` module.

16.3 Vibrationally averaged spin–spin coupling constants

For the calculation of vibrational corrections in indirect spin–spin coupling constants, DALTON provides an alternative approach to the calculation of vibrational corrections, in which also the full point-group symmetry of the molecule is exploited in order to reduce the number of displaced geometries that need to be included. An example of such an input is :

```
**DALTON INPUT
.NMDDRV
**NMDDRV
.SYMMETRY
  C2v
.VIBANA
.DISPLA
 0.01
*PROPAV
.ANHA-P
**WAVE FUNCTIONS
.HF
*SCF INPUT
.THRESH
 1.0D-12
**PROPERTIES
.SPIN-SPIN
*SPIN-S
.SELECT
  3
  1  2  3
*TRPRSP
.THRESH
1.0D-12
```

```
*LINRES
.THRESH
1.0D-12
*END OF INPUT
```

We note that the input includes the full point group symmetry of the molecule (in this case the water molecule, and thus here given as C_{2v}). In this case, the vibrational corrections are evaluated at the equilibrium geometry of the molecule, and both the harmonic and anharmonic contributions are included. The approach is very similar to that presented in the previous two sections, and a detailed description of the two approaches and a comparison of the two methods are given in Ref. [**?**]. For information about the special input modules used in the above input example, we refer to Sec. 21.2.

Chapter 17

Relativistic Effects

The following approaches to treat relativistic effects are available in DALTON:

ECP The Effective Core Potential approach of Pitzer and Winter [102] is available for single-point calculations by asking for ECP as the basis set for the chosen element. So far, only a limited set of elements is covered by the basis set library. See the `rsp_ecp` example in the test-suite. The corresponding spin-orbit operators are not implemented.

> **Reference literature:**
>
> R. M. Pitzer and N. M. Winter. *Int. J. of Quantum Chem.*, **40**, 773 (1991)
>
> L. E. McMurchie and E. R. Davidson. *J. Comp. Phys.*, **44**, 289 (1981)

Douglas-Kroll The Douglas–Kroll scalar relativistic one-electron integrals are available by adding the `.DOUGLAS-KROLL` keyword

```
**DALTON INPUT
.DOUGLAS-KROLL
.RUN WAVE FUNCTIONS
....
```

See also the `energy_douglaskroll` example in the test suite.

NOTE: Exact analytical gradients and Hessians are not available at the moment, the approximate gradient and Hessians does, however, give fairly accurate geometries. For this approach, only basis sets should be used where the contraction coefficients were optimized including the Douglas-Kroll operators. DALTON currently provides: DK-Pol (relativistic version of Sadlej's POL basis sets), raf-r for some heavy elements,

and the relativistically recontracted correlation-consistent basis sets of Dunning (cc-pVXZ-DK, X=D,T,Q,5). The combination with property operators should be done with care, *e.g.* the standard magnetic property operators are not suitable in this case.

Reference literature:

M. Douglas and N. M. Kroll. *Ann. Phys. (N.Y.)*, **82**, 89 (1974)

B. A. Hess. *Phys. Rev.* **A**, **33**, 3742 (1986)

`Spin-orbit Mean-Field` The spin-orbit mean-field approach can be used for either replacing the Breit-Pauli spin-orbit operator, or as an operator with suitable relativistic corrections in combination with the Douglas-Kroll approach. It is based on an effective one-electron operator, where the two-electron terms are summed in a way comparable to the Fock operator [103]. As all multi-center integrals are neglected, this scheme is very fast, avoids the storage of the two-electron spin-orbit integrals, and can therefore be used for large systems.

```
.....
**INTEGRALS
.MNF-SO    replaces     .SPIN-ORBIT
.....
```

For properties, the same substitution should be made, in the case of special components, `X1SPNORB` labels are replaced by `X1MNF-SO` and so on, whereas the two-electron terms will be skipped completely. For calculating phosphorescence with the quadratic response scheme, `.PHOSPHORESENCE` should be just replaced by `.MNFPHO` which takes care of choosing the appropriate integrals.

Reference literature:

B. A. Hess, C. M. Marian, U. Wahlgren and O. Gropen. *Chem. Phys. Lett.*, **251**, 365 (1996)

NOTE:

The choice between the Breit-Pauli or Douglas-Kroll mean-field operator is done by (not) providing the .DOUGLAS-KROLL keyword. It is therefore not possible to combine *e.g.* non-relativistic wave-functions with the Douglas-Kroll spin-orbit integrals.

In the present implementation, the mean-field approach works only for basis sets with a generalized contraction scheme such as the ANO basis sets, raf-r, or cc-pVXZ(-DK). For other types of basis sets, the program might work without a crash, but it will most likely provide erroneous results.

Chapter 18

SOPPA and SOPPA(CCSD) calculations

The Dalton program system can also be used to perform Second-Order Polarization Propagator Approximation (SOPPA) calculations [49, 51, 62, 63] or Second-Order Polarization Propgator Approximation with Coupled Cluster Singles and Doubles Amplitudes [SOPPA(CCSD)] calculations of optical properties like singlet or triplet excitation energies and oscillator strenghts as well as the following list of electric and magnetic properties

polarizability

magnetizability

rotational g tensor

nuclear magnetic shielding constant

nuclear spin–rotation constant

indirect nuclear spin–spin coupling constant

as well as all the linear response functions described in chapter 12.

18.1 General considerations

The Second-Order Polarization Propagator Approximation is a generalization of the SCF linear response function [49, 50, 51]. In SOPPA, the SCF reference wave function in the linear response function or polarization propagator is replaced by a Møller-Plesset wave function and all matrix elements in the response function are then evaluted through second order in the fluctuation potential. This implies that electronic excitation energies and

125

oscillator strenghts as well as linear response functions are correct through second order. Although it is a second-order method like MP2, the SOPPA equations differ significantly from the expressions for second derivatives of an MP2 energy.

In the Second Order Polarization Propgator Approximation with Coupled Cluster Singles and Doubles Amplitudes [SOPPA(CCSD)], the Møller-Plesset correlation coefficients are replaced by the corresponding coupled cluster singles and doubles amplitudes [53, 52, 63, 60]. SOPPA(CCSD) is not a coupled cluster linear response function as they are implemented in the CC modules of Dalton program. However, the equations are essentially the same as for SOPPA.

18.2 Input description

<div style="border:1px solid">

Reference literature:

General reference : E. S. Nielsen, Poul Jørgensen, and Jens Oddershede. *J. Chem. Phys.*, **73**, 6238, (1980)

General reference : J. Oddershede, Poul Jørgensen and Danny Yeager, *Comput. Phys. Rep.*, **2**, 33, (1984)

Excitation energy : Martin J. Packer, Erik K. Dalskov, Thomas Enevoldsen, Hans Jørgen Aagaard Jensen and Jens Oddershede, *J. Chem. Phys.*, **105**, 5886, (1996)

Rotational *g* tensor : Stephan P. A. Sauer, *Chem. Phys. Lett.* **260**, 271, (1996)

Polarizability : Erik K. Dalskov and Stephan P. A. Sauer. *J. Phys. Chem.* **A**, **102**, 5269, (1998)

Spin-Spin Coupling Constants : Thomas Enevoldsen, Jens Oddershede, and Stephan P. A. Sauer. *Theor. Chem. Acc.*, **100**, 275, (1998)

CTOCD-DZ nuclear shieldings: A.Ligabue, S.P.A.Sauer, P.Lazzeretti. *J.Chem.Phys.*, **118**, 6830, (2003).

</div>

A prerequisite for any SOPPA calculation is that the calculation of the MP2 energy and wavefunction is invoked by the keyword `.MP2` in the `**WAVE FUNCTIONS` input module. Furthermore in the `**PROPERTIES` or `**RESPONSE` input modules it has to be specified by the keyword `.SOPPA` that a SOPPA calculation of the properties should be carried out.

A typical input file for a SOPPA calculation of the indirect nuclear spin-spin coupling constants of a molecule will be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
```

```
.HF
.MP2
**PROPERTIES
.SOPPA
.SPIN-S
**END OF DALTON INPUT
```

whereas as typical input file for the calculation of triplet excitation energies with the **RESPONSE module will be:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
.MP2
**RESPONSE
.TRPFLG
.NOITRA
.SOPPA
*LINEAR
.SINGLE RESIDUE
.ROOTS
 4
**END OF DALTON INPUT
```

The `.NOITRA` keyword has the effect that the transformation of the two electron integrals necessary for a MP2 and SOPPA calculation is only perfomed once in the **WAVE FUNCTIONS module.

A prerequisite for any SOPPA(CCSD) calculation is that the calculation of the CCSD amplitudes for the SOPPA program is invoked by the keyword `.CC` in the **WAVE FUNCTIONS input module together with the `.SOPPA(CCSD)` option in the *CC INPUT section. Furthermore, in the **PROPERTIES or **RESPONSE input modules it has to specified by the keyword `.SOPPA(CCSD)` that a SOPPA(CCSD) calculation of the properties should be carried out.

A typical input file for a SOPPA(CCSD) calculation of the indirect nuclear spin-spin coupling constants of a molecule will be:

```
**DALTON INPUT
.RUN PROPERTIES
**WAVE FUNCTIONS
```

```
.HF
.CC
*CC INPUT
.SOPPA(CCSD)
**PROPERTIES
.SOPPA(CCSD)
.SPIN-S
**END OF DALTON INPUT
```

whereas as typical input file for the calculation of triplet excitation energies with the **RESPONSE module will be:

```
**DALTON INPUT
.RUN RESPONSE
**WAVE FUNCTIONS
.HF
.CC
*CC INPUT
.SOPPA(CCSD)
**RESPONSE
.TRPFLG
.SOPPA(CCSD)
*LINEAR
.SINGLE RESIDUE
.ROOTS
 4
**END OF DALTON INPUT
```

Chapter 19

NEVPT2 calculations

19.1 General considerations

NEVPT2 is a form of second-order multireference perturbation theory which can be applied to CAS–SCF wavefunctions or, more generally, to CAS–CI wavefunctions. The term NEVPT is an acronym for *"n–electron valence state perturbation theory"*. While we refer the reader to the pertinent literature [104, 105, 106, 107], we limit ourselves to recalling here that the most relevant feature of NEVPT2 consists in that the first order correction to the wave function is expanded over a set of properly chosen *multireference* functions which correctly take into consideration the two–electron interactions occurring among the active electrons. Among the properties ensured by NEVPT2 we quote:

- Strict separability (size consistence): the energy of a collection of non–interacting systems equals the sum of the energies of the isolated systems

- Absence of intruder states: the zero-order energies associated to the functions of the outer space are well separated from the zero-order energy of the state being studied, thus avoiding divergences in the perturbation summation

- The first order correction to the wavefunction is an eigenfunction of the spin operators S^2 and S_z

- Electronically excited states are dealt with at the same level of accuracy as the ground state

- NEVPT2 energies are invariant under a unitary transformation of the active orbitals. Furthermore, the choice of canonical orbitals for the core and virtual orbitals (the default choice) ensure that the results coincide with those of an enlarged version of the theory fully invariant under rotations in the core and virtual orbital spaces, respectively [107]

- NEVPT2 coincides with MP2 in the case of a HF wave function

NEVPT2 has been implemented in two variants both of which are present in DALTON, these are the *strongly contracted* (SC) and the *partially contracted* (PC) variants. The two variants differ by the number of perturber functions employed in the perturbation summation. The PC–NEVPT2 uses a richer function space and is in general more accurate than the SC–NEVPT2. The results of SC–NEVPT2 and PC–NEVPT2 are anyway usually very close to one another.

Reference literature:

General reference : C. Angeli, R. Cimiraglia, S. Evangelisti, T. Leininger and J. P. Malrieu, *J. Chem. Phys.*, **114**, 10252, (2001)

General reference : C. Angeli, R. Cimiraglia and J. P. Malrieu, *Chem. Phys. Lett.*, **350**, 297, (2001)

General reference : C. Angeli, R. Cimiraglia and J. P. Malrieu, *J. Chem. Phys.*, **117**, 9138, (2002)

Excited states : C. Angeli, S. Borini and R. Cimiraglia, *Theor. Chem. Acc.*, **111**, 352 (2004)

19.2   Input description

NEVPT2 must follow a CAS–SCF or CAS–CI calculation. The keyword `.NEVPT2` has to be specified in the `**WAVE FUNCTIONS` data section. Furthermore a small `*NEVPT2` data group can be specified providing the few input data that can optionally be provided by the user: `.THRESH`, the threshold to discard small coefficients in the CAS wavefunction (default = 0.0), `.FROZEN`, a vector specifying for each symmetry the core orbitals which are excluded from the correlation treatment (the default is no freezing) and `.STATE`, the state in a CASCI calculation. This keyword is unnecessary (ignored) in the CASSCF case. An example of a NEVPT2 calculation is

```
**DALTON INPUT
**WAVE FUNCTIONS
.MCSCF
.NEVPT2
*NEVPT2
.THRESH
1.0D-12
.FROZEN
 1 0 1 0
**END OF DALTON INPUT
```

Chapter 20

Examples of coupled cluster calculations

We collect in this Chapter a few examples of input files (`DALTON.INP`) for calculations one might want to carry out with the CC modules of the Dalton program. Other examples may be found in the test suite.

It should be stressed that all modules in CC can be used simultaneously—assuming they are all implemented for the chosen wavefunction model(s). For instance, linear, quadratic and cubic response functions can be obtained within the same calculation.

20.1 Multiple model energy calculations

CC allows for the calculation of the ground state energy of the given system using a variety of wavefunction models, listed in Section 27.1. For any model specified, the Hartree–Fock energy is always calculated. The following input describes the calculation of SCF, MP2, CCSD and CCSD(T) ground state energies:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.CC
*CC INPUT
.MP2
.CCSD
.CC(T)
**END OF DALTON INPUT
```

Note that SCF, MP2 and CCSD energies are obtained by default if CCSD(T) (`.CC(T)`)

is required. Therefore the keywords `.MP2` and `.CCSD` may also be omitted in the previous example.

20.2 First-order property calculation

The following input exemplifies the calculation of all orbital-relaxed first-order one-electron properties available in cc, for the hierarchy of wavefunction models SCF (indirectly obtained through CCS), MP2, CCSD and CCSD(T). For details, see Section 27.2.

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**INTEGRALS
.DIPLEN
.SECMOM
.THETA
.EFGCAR
.DARWIN
.MASSVELO
**WAVE FUNCTIONS
.CC
*CC INPUT
.CCS            (gives SCF First order properties)
.MP2            (default if CCSD is calculated)
.CCSD
.CC(T)
*CCFOP
.ALLONE
**END OF DALTON INPUT
```

20.3 Static and frequency-dependent dipole polarizabilities and corresponding dispersion coefficients

The following input describes the calculation of the electric dipole polarizability component $\alpha_{zz}(\omega)$, for $\omega = 0.00$ and $\omega = 0.072$ au, and its dispersion coefficients up to order 6, in the hierarchy of CC models CCS, CC2 and CCSD. The calculation of the electric dipole moment has been included in the same run.

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
```

```
**INTEGRALS
.DIPLEN
**WAVE FUNCTIONS
.CC
*CC INPUT
.CCS
.CC2
.CCSD
*CCFOP
.DIPMOM
*CCLR
.OPERATOR
ZDIPLEN ZDIPLEN
.FREQUENCIES
  2
0.00  0.072
.DISPCF
  6
**END OF DALTON INPUT
```

20.4 Static and frequency-dependent dipole hyperpolarizabilities and corresponding dispersion coefficients

The previous input can be extended to include the calculation of the electric first and second hyperpolarizability components $\beta_{zzz}(\omega_1, \omega_2)$ and $\gamma_{zzzz}(\omega_1, \omega_2, \omega_3)$, for $\omega_1 = \omega_2 = (\omega_3 =) 0.00$ and $\omega_1 = \omega_2 = (\omega_3 =) 0.072$ au. The corresponding dispersion coefficients up to sixth order are also calculated. For other specific cases, see Sections 27.4 and 27.5

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**INTEGRALS
.DIPLEN
**WAVE FUNCTIONS
.CC
*CC INPUT
.CCS
.CC2
.CCSD
*CCFOP
```

```
.DIPMOM
*CCLR
.OPERATOR
ZDIPLEN ZDIPLEN
.FREQUENCIES
  2
0.00  0.072
.DISPCF
  6
*CCQR
.OPERATOR
ZDIPLEN ZDIPLEN ZDIPLEN
.MIXFRE
  2
0.00  0.072                    !omega_1
0.00  0.072                    !omega_2
*CCCR
.OPERATOR
ZDIPLEN ZDIPLEN ZDIPLEN ZDIPLEN
.MIXFRE
  2
0.00  0.072                    !omega_1
0.00  0.072                    !omega_2
0.00  0.072                    !omega_3
.DISPCF
 6
**END OF DALTON INPUT
```

Obviously, linear, quadratic and cubic response modules can also be run separately.

20.5 Excitation energies and oscillator strengths

This is an example for the calculation of singlet excitation energy and oscillator strength for a system with NSYM = 2 (C_s).

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**INTEGRAL
.DIPLEN
```

```
**WAVE FUNCTIONS
.CC
*CC INPUT
.CCSD
.NSYM
 2
*CCEXCI
.NCCEXCI                 !number of excited states
 2 1                     !2 states in symmetry 1 and 1 state in symmetry 2
*CCLRSD
.DIPOLE
**END OF DALTON INPUT
```

Triplet excitation energies can be obtained adding an extra line to `.NCCEXCI` specifying the number of required triplet excited states for each symmetry class. Note however that linear residues are not available for triplet states (The `*CCLRSD` sections should be removed).

20.6 Gradient calculation, geometry optimization

Available for CCS, CC2, MP2, CCSD and CCSD(T) using integral direct analytic gradient. For a single integral-direct gradient calculation:

```
**DALTON INPUT
.DIRECT
.RUN WAVE FUNCTIONS
**INTEGRAL
.DIPLEN
.DEROVL
.DERHAM
**WAVE FUNCTIONS
.CC
*CC INP
.CCSD
*DERIVATIVES
**END OF DALTON INPUT
```

Note that if several wavefunction models are specified, the gradient calculation is performed only for the "lowest-level" model in the list.
For a geometry optimization:

```
**DALTON INPUT
.OPTIMIZE
**INTEGRAL
.DIPLEN
.DEROVL
.DERHAM
**WAVE FUNCTIONS
.CC
*CC INPUT
.CC(T)
**END OF DALTON INPUT
```

20.7   R12 methods

At present available at the MP2 level. The input for an MP2-R12/A calculation is as follows (the key `.AUXBAS` is used only if an auxiliary basis is employed):

```
**DALTON INPUT
.DIRECT
.RUN WAVE FUNCTIONS
**INTEGRAL
.R12
.AUXBAS
**WAVE FUNCTIONS
.CC
*CC INPUT
.MP2
*R12
.NO A'
.NO B
**END OF DALTON INPUT
```

Part III

DALTON Reference Manual

Chapter 21

General input module

In this chapter the general structure of the input file for DALTON is described, as well as the possible input cards that can be entered in the **DALTON input module. This input section should always begin an input file for the DALTON program.

21.1 General input to DALTON : **DALTON

This input module describes the overall type of calculation that is to be done. It also contains four submodules describing the performance of parallel calculations, the control of the two different geometry optimization routines, as well as control the general routines for calculating numerical geometrical derivatives of molecular energies or selected first- and second-order properties. We note that this input module has to start all input files for DALTON.

.DIRECT The calculation is to be done in a direct manner, that is, the two-electron integrals are to be constructed "on the fly" and not written to disc as is the default. This keyword will only work for SCF wave functions, Density Functional Theory calculation and for coupled cluster (CC) calculations. In HF and DFT calculations the two-electron integrals (and differentiated two-electron integrals) will not be written to disc in any part of the calculation, whereas in the direct CC approach, the two-electron integrals will be stored in three general-indexed batches, thus requiring a loop over all basis functions for one of the two-electron integral indices [?].

.DOUGLAS-KROLL Include scalar relativistic effects by using the Douglas-Kroll transformed one-electron potential and kinetic energy Hamiltonian.

.INPTES Test the input of the **DALTON INPUT input module. The program will abort after the completion of the input test, and no calculation will be executed.

.INTEGRALS Invoke the HERMIT and/or the ERI program for generating molecular one-
and two-electron integrals. See Chapter 22 for the HERMIT and Section 22.2.8 ERI
programs, respectively.

.ITERATION
      READ (LUCMD, '(I5)') ITERNR

Tells the program at which iteration to start the geometry optimization using the
**WALK module. Note that this will *not* affect which molecule input file that is going
to be read, as this is handled by the job script `dalton`. It only determines what
number the output of the predicted molecular geometry will be, as well as where to
start writing information on the files containing information about an IRC calculation
(`DALTON.IRC`) or a dynamical trajectory walk (`DALTON.TRJ`).

.MAX IT
      READ (LUCMD, '(I5)') ITERMX

Change the maximum number of geometry iterations that can be done. Default is 20.
For numerical differentiation/vibrational averaging, the number of iterations will be
reset to $6N + 1$ (where N is the number of nuclei) as the number of required iterations
for these calculations are well defined. This number has to be increased in Intrinsic
Reaction Coordinate (IRC) or dynamical trajectory studies. However, changing this
variable will override this reset option.

.OPTIMIZE Do a geometry walk. If no input is given in the *OPTIMIZE input submodule, an
optimization of the molecular geometry to a stationary point with no negative Hessian
eigenvalues (a local minimum) will be done using the default first-order methods. How-
ever, this may be changed using appropriate keywords in the submodule *OPTIMIZE,
and we refer to examples in the chapter on potential energy surfaces (Chapter 7), and
subsection 21.1.3 describing the input cards for the *OPTIMIZE submodule for a more
detailed description of possible options.

.NMDDRV Calls for a generalized numerical geometry differentiation. These routines will take
advantage of the full molecular point group in order to minimize the number of point
to be calculated. What order of derivatives and whether any analytical derivatives
are to be used is determined in the *NMDDRV input module.

.PARALLEL Requests that the calculation of two-electron integrals is to be done in parallel.
This also implies that the calculation is done without writing two-electron integrals to
disc. This keyword only applies to SCF wave functions and DFT calculations, but all
two-electron integral evaluations in an SCF calculation will be done parallel as well

as the integration of the exchange-correlation functionals and kernels. More details about the parallelization strategy in DALTON can be found in Ref. [90].

The keyword requires that the program has been installed and compiled with the appropriate preprocessor directives for an MPI installation. Note that in order to evaluate the parallelization efficiency, a print level of at least 2 is needed in the `*PARALLEL` submodule.

.PARNMD Option non-functional. Do not use.

.PRESORT Requests that the two-electron integrals should be sorted and that the integral transformation routines of Bjørn Roos should be used during execution of the program. The keyword is needed if one attempts an MCSCF, CI or MP2 (run through the SIRIUS module and not using the Coupled-Cluster module) with more than 255 basis functions.

.PRIERR
READ (LUCMD, *) IPRERR Reads in the print level that is to be used in the DALTON.ERR file. Default print level is `IPRUSR+1`.

.PRINT
READ (LUCMD, *) IPRUSR Reads in the print level that is to be used the rest of the subsequent calculations. Default is a print level of 0.

.PROPERTIES Invoke the ABACUS program for the evaluation of static and dynamic properties. See Chapter 25.

.RESPONSE Invoke the RESPONSE program for the evaluation of static and dynamic properties. See Chapter 26.

.RUN ALL Invoke all the programs HERMIT, SIRIUS, RESPONSE, and ABACUS for a single point calculation.

.RUN PROPERTIES Invoke the programs HERMIT, SIRIUS, and ABACUS for a single point calculation.

.RUN RESPONSE Invoke all the programs HERMIT, SIRIUS, and RESPONSE for a single point calculation.

.RUN WAVE FUNCTIONS Invoke the programs HERMIT and SIRIUS for a single point energy calculation.

.RUNERI Force the use of the vectorized integral code ERI where possible.

.TOTSYM Consider only totally symmetric perturbations. This option only affects geometric perturbations calculated using the second-order based .WALK option and static electric-field perturbations requested through the keyword .POLARI.

.VECLEN

    READ(LUCMD,*) IVECLN

Set the number of Fock matrices to be used during Fock-matrix constructions in direct calculations. This function is only of interest for vector machines. The default is 128. The larger the number, the more memory will be required in the calculation.

.WALK Do a geometry walk. If no input is given in the *WALK input submodule, an optimization of the molecular geometry to a stationary point with no negative Hessian eigenvalues (a local minimum) will be done using a second-order method with analytical Hessians. However, this may be changed by appropriate keywords in the submodule *WALK, and we refer to examples in the chapter on potential energy surfaces (Chapter 7), and subsection 21.1.3 describing the input cards for the *WALK submodule for a more detailed description of possible options.

.WAVE FUNCTIONS Invoke the SIRIUS program for the evaluation of SCF, MP2, Coupled Cluster and MCSCF wave functions as well as DFT calculations. See Chapter 24.

21.1.1 General: *OPTIMIZE

This submodule is the driver for geometry optimizations, although we note that the *WALK module contains another second-order geometry optimization driver. The *OPTIMIZE module contains both first and second-order methods for locating minima and transition states (geometry optimization). Most of the Hessian updating schemes were taken from ref.[108] and [23]. The implementation of redundant internal coordinates follows the work of Peng *et al.* [109]. In addition to this, several keywords for VRML visualization are included [110].

.1STORD Use default first-order method. This means that the BFGS update will be used, and that the optimization is carried out in redundant internal coordinates. Same effect as the combination of the two keywords .BFGS and .REDINT. Since the .BFGS method ensures a positive definite Hessian, the .BOFILL optimization method is used by default in case of searches for transition states.

.2NDORD Use default second-order method. Molecular Hessians will be calculated at every geometry. The level-shifted Newton method and Cartesian coordinates are used. Identical to specifying the keywords .NEWTON and .CARTES.

.BAKER Activates the convergence criteria of Baker [34]. The minimum is then said to be found when the largest element of the gradient vector (in Cartesian or redundant

internal coordinates) falls below $3.0 \cdot 10^{-4}$ and either the energy change from the last iteration is less than $1.0 \cdot 10^{-6}$ or the largest element of the predicted step vector is less $3.0 \cdot 10^{-4}$.

.BFGS Specifies the use of a first-order method with the Broyden-Fletcher-Goldfarb-Shanno (BFGS) update formula for optimization. This is the preferred first-order method for minimizations, as this update is able to maintain a positive definite Hessian. Note that this also makes it unsuitable for transitions state optimization (where one negative eigenvalue is sought).

.BFGSR1 Use a linear combination of the BFGS and the symmetric rank one updating schemes in the same fashion as Bofill's update. Only suitable for minimizations.

.BOFILL Bofill's update[35] is the default updating scheme for transition state optimizations. It's a linear combination of the symmetric rank one and the PSB updating schemes, automatically giving more weight to PSB whenever the rank one potentially is numerically unstable.

.CARTES Indicates that Cartesian coordinates should be used in the optimization. This is the default for second-order methods.

.CMBMOD Uses a combination of the BFGS update and the model Hessian (diagonal in redundant internal coordinates). The two have equal weight in the first iteration of the geometry optimization, then for each subsequent iteration the weight of the model Hessian is halved. Only suitable for minimizations.

.CONDIT

```
READ (LUCMD,*) ICONDI
```

Set the number of convergence criteria that should be fulfilled before convergence occurs. There are three different convergence thresholds, one for the energy, one for the gradient norm and one for the step norm. The possible values for this variable is therefore between 1 and 3. Default is 2. The three convergence thresholds can be adjusted with the keywords **.ENERGY**, **.GRADIE** and **.STEP T**.

.CONSTRAINT

```
READ (LUCMD, *) NCON
DO I = 1, NCON
   READ(LUCMD,*) ICON
   ICNSTR(ICON) = 1
END DO
```

Request a constrained geometry optimization. Only works when using redundant internal coordinates. The number of primitive coordinates that should be frozen has to be specified (`NCON`), then a list follows with the individual coordinate numbers. The coordinate numbers can be found by first running DALTON with the `.FINDRE` keyword. Any number of bonds, angles and dihedral angles may be frozen. NOTE: Symmetry takes precedence over constraints, if you *e.g.* want to freeze just one of several symmetric bonds, symmetry must be lowered or switched off.

`.DELINT` Use delocalized internal coordinates. These are built up as non-redundant linear combinations of the redundant internal coordinates. Performance is more or less the same as for the redundant internals, but the transformation of displacements (step) is slightly less stable.

`.DFP` Specifies that a first-order method with the Davidon-Fletcher-Powell (DFP) update formula should be used for optimization. May be used for both minimizations and transition state optimizations.

`.DISPLA`

    `READ (LUCMD,*) DISPLA`

Read one more line containing the norm of the displacement vector to be used during numerical evaluation of the molecular gradient, as is needed when doing geometry optimizations with CI or MP2 wave functions. Default is $1.0 \cdot 10^{-3}$ a.u.

`.ENERGY`

    `READ(LUCMD,*) THRERG`

Set the convergence threshold for the energy. This is one of the three convergence thresholds (the keywords `.GRADIE` and `.STEP T` control the other two). Default value is the maximum of $1.0 \cdot 10^{-6}$ and two times the threshold for the wave function gradient.

`.FINDRE` Determines the redundant internal coordinate system then quits without doing an actual calculation. Useful for setting up constrained geometry optimizations, where the numbers of individual primitive internal coordinates are needed.

`.GDIIS` Use the Geometrical DIIS[111] algorithm to control the step. Works in much the same way as DIIS for wave functions. However, the rational function and level-shifted Newton methods are generally more robust and more efficient. Can only be used for minimizations.

`.GEOANA` Enables an analysis of the molecular geometry in terms of bond lengths and bond angles at each new geometry predicted during the optimization procedure.

`.GRADIE`

> `READ(LUCMD,*) THRGRD`

> Set the convergence threshold for the gradient norm. This is one of the three convergence thresholds (the keywords `.ENERGY` and `.STEP T` control the other two). Default value is the maximum of $1.0 \cdot 10^{-5}$ and two times the threshold for the wave function gradient.

`.GRDINI` Specifies that the Hessian should be reinitialized every time the norm of the gradient is larger than norm of the gradient two iterations earlier. This keyword should only be used when it's difficult to obtain a good approximation to the Hessian during optimization. Only applies to first-order methods.

`.HELLMA` Use gradients and Hessians calculated using the Hellmann-Feynman approximation. Currently not working properly

`.HESFIL` Specifies that the initial Hessian should be read from the file `DALTON.HES`. This applies to first-order methods, and the Hessian in the file must have the correct dimensions. This option overrides other options for the initial Hessian.

> Each time a Hessian is calculated or updated, it's written to this file (in Cartesian coordinates). If an optimization is interrupted, it can be restarted with the last geometry and the Hessian in `DALTON.HES`, minimizing the loss of information. Another useful possibility, is to transfer the Hessian from a calculation on the same molecule with another (smaller) basis and/or a cheaper wave function. Finally, one can go in and edit the file directly to set up a specific force field.

`.INIMOD` Use a simple model Hessian [33] diagonal in redundant internal coordinates as the initial Hessian. All diagonal elements are determined based on an extremely simplified molecular mechanics model, yet this model provides Hessians that are good starting points for most systems, thus avoiding any calculation of the exact Hessian. This is the default for first-order methods.

`.INIRED` Specifies that the initial Hessian should be diagonal in redundant internal coordinates. The different diagonal elements are set equal to 0.5 for bonds, 0.2 for angles and 0.1 for dihedral angles, unless `.INITEV` has been specified. If the optimization is run in Cartesian coordinates, the diagonal internal Hessian is transformed to Cartesians. Only applies to first-order methods.

`.INITEV`

> `READ(LUCMD,*) EVLINI` The default initial Hessian for first-order minimizations is the identity matrix when Cartesian coordinates are used, and a diagonal matrix when

redundant internal coordinates are used. If `.INITEV` is used, all the diagonal elements (and therefore the eigenvalues) are set equal to the value EVLINI. This option only has effect when first-order methods are used and `.INITHE` and `.HESFIL` are non-present.

`.INITHE` Specifies that the initial Hessian should be calculated (analytical Hessian), thus yielding a first step that is identical to that of second-order methods. This provides an excellent starting point for first-order methods, but should only be used when the Hessian can be calculated within a reasonable amount of time. It has only effect for first-order methods and overrides the keywords `.INITEV` and `.INIRED`. It has no effect when `.HESFIL` has been specified.

`.LINE S` Turns on line searching, using a quartic polynomial. By default this is turned off, as there seems to be no gain in efficiency. Can only be used for minimizations.

`.M-BFGS` A list of old geometries and gradients are kept. At each new point, displacements and gradient difference for the last few steps are calculated, and all of these are then used to sequentially update the Hessian, the most weight being given to the last displacement and gradient difference. Each update is done using the BFGS formula, and it's thus only suitable for minimizations. Only applies to first-order methods.

`.M-PSB` This identical to `.M-BFGS`, except the PSB formula is used for the updating. Only applies to first-order methods, but it can be used for both minimizations and saddle point optimizations.

`.MAX IT`

    `READ(LUCMD,*) ITRMAX`

Read the maximum number of geometry iterations. Default value is 25.

`.MAX RE`

    `READ(LUCMD,*) MAXREJ`

Read maximum number of rejected steps in each iterations, default is 3.

`.MODE`

    `READ(LUCMD,*) NSPMOD`

Only has effect when doing saddle point optimizations. Determines which Hessian eigenmode should be maximized (inverted in the image method). By default this is the mode corresponding to the lowest eigenvalue, *i.e.* mode 1. If an optimization does not end up at the correct transition state, it may be worthwhile following other modes (only the lower ones are usually interesting).

.MODHES Determine a new model Hessian (see `.INIMOD`) at every geometry without doing any updating. The model is thus used in much the same manner as an exact Hessian, though it is obviously only a relatively crude approximation to the analytical Hessian.

.NEWTON Specifies that a second-order Newton method should be used for optimization—that is, the analytical Hessian will be calculated at every geometry. By default the level-shifted trust region method will be used, but it is possible to override this by using one of the two keywords `.RF` or `.GDIIS`.

.NOAUX Only has effect when using redundant internal coordinates. The default for minimizations is to add auxiliary bonds between atoms that are up to two and half times further apart then regular (chemical) bonds. This increases the redundancy of the coordinate system, but usually speeds up the geometry optimization slightly. `.NOAUX` turns this off. For saddle point optimizations and constrained geometry optimization this is off by default (cannot be switched on).

.NOBREA Disables breaking of symmetry. The geometry will be optimized within the given symmetry, even if a non-zero Hessian index is found. The default is to let the symmetry be broken until a minimum is found with a Hessian index of zero. This option only has effect when second-order methods are used.

.NODIHE Only has effect when using redundant internal coordinates. Removes all coordinates that are dihedral angles, leaving only bonds and angles. Not too useful, but may be used if one wants to limit the number of internal coordinates. Constrained geometry optimizations can sometimes benefit from having all dihedral angles removed (assuming no dihedral angles needs to be frozen).

.NOTRUS Turns off the trust radius, so that a full Newton step is taken in each iteration. This should be used with caution, as global convergence is no longer guaranteed. If long steps are desired, it is safer to adjust the initial trust radius and the limits for the actual/predicted energy ratio.

.PREOPT

```
READ (LUCMD,*) NUMPRE
DO I = 1, NUMPRE
   READ (LUCMD,*) PREBTX(I)
END DO
```

First we read the number of basis sets that should be used for preoptimization, then we read those basis set names as strings. These sets will be used for optimization in the order they appear in the input. One should therefore place the smaller basis at

the top. After the preoptimization, optimization is performed with the basis specified in the molecule input file.

.PRINT

    `READ (LUCMD,*) IPRINT`

Set print level for this module. Read one more line containing print level. Default value is 0, any value higher than 12 gives debugging level output.

.PSB Specifies that a first-order method with the Powell-Symmetric-Broyden (PSB) update formula should be used for optimization.

.RANKON Specifies that a first-order method with the rank one update formula should be used for optimization. This updating is also referred to as symmetric rank one (SR1) or Murtagh-Sargent (MS).

.REDINT Specifies that redundant internal coordinates should be used in the optimization. This is the default for first-order methods.

.REJINI Specifies that the Hessian should be reinitialized after every rejected step, as a rejected step indicates that the Hessian models the true potential surface poorly. Only applies to first-order methods.

.REMOVE

    `READ (LUCMD, *) NREM`
    `DO I = 1, NREM`
       `READ(LUCMD,*) IREM`
       `ICNSTR(IREM) = 2`
    `END DO`

Only has effect when using redundant internal coordinates. Specifies internal coordinates that should be removed. The input is identical to the one for `.CONSTRAINT`, that is one has to specify the number of coordinates that should be removed, then the number of each of those internal coordinates. The coordinate numbers can first be determined by running with `.FINDRE` set.

Removing certain coordinates can sometimes be useful in speeding up constrained geometry optimization, as certain coordinates sometimes "struggle" against the constraints. See also `.NODIHE`.

.RF Use the rational function method [27] instead of level-shifted Newton which is the default. The RF method is often slightly faster than the level-shifted Newton, but also slightly less robust.

For saddle point optimizations there's a special partitioned rational function method (used automatically when both `.RF` and `.SADDLE` are set). However, this method is both slower and less stable than the default trust-region level-shifted image method (which is the default).

`.SADDLE` Indicates that a saddle point optimization should be performed rather than a minimization. The default method is to calculate the Hessian analytically at the initial geometry, then update it using Bofill's update. The optimization is performed in redundant internal coordinates and using the trust-region level-shifted image method to control the step. That is by default all the keywords `.INITHE`, `.BOFILL` and `.REDINT` are already set, but this can of course be overridden by specifying other keywords. If locating the desired transition state is difficult, and provided analytical Hessians are available, it may sometimes be necessary to use the `.NEWTON` keyword so that Hessians are calculated at every geometry.

`.SCHLEG` Specifies that a first-order method with Schlegel's updating scheme [112] should be used. This makes use of all previous displacements and gradients, not just the last, to update the Hessian.

`.SP BAS`
> `READ(LUCMD,*) SPBSTX`

Read a string containing the name of a basis set. When the geometry has converged, a single-point energy will be calculated using this basis set.

`.STABILIZE`
> `READ(LUCMD,*) ISTBLZ`

Tries to "stabilize" the predicted new molecular geometries (and thus reduce the risk of symmetry breakings) by ignoring all numbers appearing in the Cartesian coordinates of the atoms beyond digit number `ISTBLZ`.

`.STEEPD` Specifies that the first-order steepest descent method should be used. No update is done on the Hessian, so the optimization will be guided by the gradient alone. The "pure" steepest descent method is obtained when the Hessian is set equal to the identity matrix. Each step will then be the negative of the gradient vector, and the convergence towards the minimum will be extremely slow. However, this option can be combined with other initial Hessians in Cartesian or redundant internal coordinates, giving a method where the main feature is the lack of Hessian updates (static Hessian).

`.STEP T`
> `READ(LUCMD,*) THRSTP`

Set the convergence threshold for the step norm. This is one of the three convergence thresholds (the keywords `.ENERGY` and `.GRADIE` control the other two). Default value is $1.0 \cdot 10^{-5}$.

.SYMTHR

    READ(LUCMD,*) THRSYM

Determines the gradient threshold for breaking of the symmetry. That is, if the index of the Hessian is non-zero when the gradient norm drops below this value, the symmetry is broken to avoid unnecessary iterations within the wrong symmetry. This option only applies to second-order methods and when the keyword `.NOBREA` is not present. The default value of this threshold is $5.0 \cdot 10^{-3}$.

.TR FAC

    READ(LUCMD,*) TRSTIN, TRSTDE

Read two factors that will be used for increasing and decreasing the trust radius respectively. Default values are 1.2 and 0.7.

.TR LIM

    READ(LUCMD,*) RTENBD, RTENGD, RTRJMN, RTRJMX

Read four limits for the ratio between the actual and predicted energies. This ratio indicates how good the step is—that is, how accurately the quadratic model describes the true energy surface. If the ratio is below `RTRJMN` or above `RTRJMX`, the step is rejected. With a ratio between `RTRJMN` and `RTENBD`, the step is considered bad an the trust radius decreased to less than the step length. Ratios between `RTENBD` and `RTENGD` are considered satisfactory, the trust radius is set equal to the norm of the step. Finally ratios above `RTENGD` (but below `RTRJMX`) indicate a good step, and the trust radius is given a value larger than the step length. The amount the trust radius is increased or decreased can be adjusted with `.TR FAC`. The default values of `RTENBD`, `RTENGD`, `RTRJMN` and `RTRJMX` are 0.4, 0.8, -0.1 and 3.0 respectively.

.TRSTRG Specifies that the level-shifted trust region method should be used to control the step. This is the default, so the keyword is actually redundant at the moment. Alternative step control methods are `.RF` and `.GDIIS`.

.TRUSTR

    READ(LUCMD,*) TRSTRA

Set initial trust radius for calculation. This will also be the maximum step length for the first iteration. The trust radius is updated after each iteration depending on the ratio between predicted and actual energy change. The default trust radius is 0.5 a.u.

.VISUAL Specifies that the molecule should be visualized, writing a VRML file of the molecular geometry. *No optimization will be performed when this keyword is given.* See also related keywords `.VR-BON`, `.VR-COR`, `.VR-EIG` and `.VR-VIB`.

.VRML Specifies that the molecule should be visualized. VRML files describing both the initial and final geometry will be written (as `initial.wrl` and `final.wrl`). The file `final.wrl` is updated in each iteration, so that it always reflects the latest geometry. See also related keywords `.VR-BON`, `.VR-COR`, `.VR-EIG` and `.VR-VIB`.

.VR-BON Only has effect together with `.VRML` or `.VISUAL`. Specifies that the VRML files should include bonds between nearby atoms. The bonds are drawn as grey cylinders, making it easier to see the structure of the molecule. If `.VR-BON` is omitted, only the spheres representing the different atoms will be drawn.

.VR-COR Draws x-, y- and z-axis in the VRML scenes with geometries. Somewhat useful if one is struggling to build a reasonable geometry by adjusting coordinates manually.

.VR-EIG Only has effect together with `.VRML` or `.VISUAL`. Specifies that the eigenvectors of the molecule (that is the eigenvectors of the Hessian, which differs from the normal modes as they are not mass-scaled) should be visualized. These are written to the files `eigv_###.wrl`.

.VR-SYM Draws in all symmetry elements of the molecule as vectors (rotational axes) and semi-transparent planes (mirror planes).

.VR-VIB Similar to `.VR-EIG`, but more useful as it draws the actual normal mode vectors (the mass-weighted eigenvectors). These are written to the files `norm_###.wrl`. Keyword only has effect when a vibrational analysis has been requested.

21.1.2   Parallel calculations : *PARALLEL

This submodule controls the performance of the parallel version of DALTON. The implementation has been described elsewhere [90]. DALTON only supports MPI as message passing interface in the current release.

.DEBUG Transfers the print level from the master to the slaves, otherwise the print level on the slaves will always be zero. Only for debugging purposes.

.DEGREE
```
READ (LUCMD,*) NDEGDI
```
Determines the percent of available tasks that is to be distributed in a given distribution of tasks, where a distribution of tasks is defined as the process of giving batches

to *all* slaves. The default is 5% , which ensures that each slave will receive 20 tasks during one integral evaluation, which will give a reasonable control with the idle time of each slave.

.NODES

> `READ (LUCMD,*) NODES`

When MPI is used as message passing interface, the default value is the number of nodes that has been assigned to the job, and these nodes will be partitioned into one master and `NODES-1` slaves. In most cases the program will find the number of nodes from the run-shell environment and setting equal to the number of nodes requested when submitting the MPI job minus 1.

.PRINT

> `READ (LUCMD, *) IPRPAR`

Read in the print level for the parallel calculation. A print level of at least 2 is needed in order to be able to evaluate the parallel efficiency. A complete timing for all nodes will be given if the print level is 4 or higher.

21.1.3 Geometry optimization: `*WALK`

Directives controlling one of the two second-order geometry optimizations as well as the execution of dynamical walks and numerical differentiation in calculations of Raman intensities and optical activity, appear in the `*WALK` section.

.ANHARM Requests that a determination of the cubic force field is to be determined. By default this is done calculating numerical derivatives of analytical Hessians in Cartesian coordinates.

.DISPLA

> `READ (LUCMD, *) DISPLC`

Displacement taken in a numerical differentiation. This applies both for a numerical molecular Hessian, as well as in calculation of Raman intensities and optical activity. Read one more line specifying value (*). Default is 10^{-4} a.u. However, note that this variable do not determine the displacements used when evaluating numerical gradient for use in first-order geometry optimizations with MP2 or CI wave functions, which is controlled by the `.DISPLA` keyword in the `*OPTIMIZE` module.

.DYNAMI Perform a "dynamic walk": integrate the classical equations of motion for the nuclei analytically on a locally quadratic surface. The method is discussed in Ref. [28] as well as in Section 7.2.2.

.ECKART

```
     DO I = 1, NUCDEP
        READ (LUCMD,'(7X,F17.10,2F24.10)')
&           (ECKGEO(J,I), J = 1, 3)
     END DO
```

During a vibrational averaging, ensure that the properties are transformed to the appropriate Eckart axis system. The coordinate system given should be that of the equilibrium geometry of the molecule. The coordinates should be given in bohr, and should be given for all symmetry generated atoms in the order given in the input.

.EIGEN Take a step to the boundary of the trust region along the eigenvector mode specified by **.MODE**.

.FRAGME

```
     READ (LUCMD, *) NIP
     READ (LUCMD, *) (IPART(IP), IP = 1, NIP)
```

Identify which fragments atoms belong to in a dynamic walk. Read one more line specifying the number of atoms (the total number of atoms in the molecule), then one more line identifying which fragment an atom belongs to. The atoms in the molecule are given a number, different for each fragment. See also the discussion in Sec. 7.2.2.

.GRDEXT Perform a gradient extremal-based optimization. The algorithm used in this kind of optimization is thoroughly described in Ref.[25]. This is the default walk type if the index of the critical point searched is higher than 1. See also the discussion in Sec. 7.1.4.

.HARMON

```
     READ (LUCMD, *) ANHFAC
```

Threshold for harmonic dominance. Read one more line specifying value. Default is 100. This is another way of changing the criterion for changes of the trust radius. See also the keyword **.TRUST**.

.IMAGE Locate a transition state using a trust-region-based image surface minimization. Note that only a point with a Hessian index of 1 can currently be located with this method, not higher-order stationary points. See also the discussion in Sec. 7.1.2.

.INDEX

```
     READ (LUCMD,*) IWKIND
```

Desired Hessian index (strictly speaking, of the totally symmetric block of the Hessian) at the optimized geometry. Read one more line specifying value. Default is 0

(minimum). Note that a stationary point with the wrong Hessian index will not be accepted as an optimized geometry.

.IRC

    READ (LUCMD, *) IRCSGN

Set the geometry walk to be an Intrinsic Reaction Coordinate (IRC) as described in Ref. [37]. Read one more line containing the sign (-1 or 1) of the reaction coordinate. It cannot be decided in advance which reaction pathway a specific sign is associated with. See also the discussion in Sec. 7.2.1.

.KEEPSY Ensure that the symmetry of the molecule is not broken. The threshold for determining a mode as breaking symmetry is controlled by the keyword **.ZERGRD**.

.MASSES Mass-scale the atomic coordinates. This is the default for dynamic walks, gradient extremal walks and in calculations of Intrinsic Reaction Coordinates (IRCs).

.MAXNUC

    READ (LUCMD, *) XMXNUC

Maximum displacement allowed for any one atom as a result of the geometry update. Read one more line specifying value. Default is 0.5.

.MAXTRU

    READ (LUCMD, *) TRUMX1

Set the maximum arc length in an Intrinsic Reaction Coordinate (IRC) walk. Read one more line containing the maximum arc length. Default is 0.10. Note that this arc length is also affected by the **.TRUST** keyword, and if both are specified, the arc length will be set to the minimum value of these to.

.MODE

    READ (LUCMD,*) IMODE

Mode to follow in level-shifted Newton optimizations for transition states. Read one more line specifying mode. Default is to follow the lowest mode (mode 1).

.MODFOL Perform a mode-following (level-shifted Newton) optimization. This is the default for minimizations and localization of transition states. See also discussion in Section 7.1.5.

.MOMENT

    READ (LUCMD, *) NSTMOM
    DO IP = 1, NSTMOM

```
        READ (LUCMD, *) ISTMOM(IP), STRMOM(IP)
    END DO
```

Initial momentum for a dynamic walk. Read one more line specifying the number of modes to which there is added an initial momentum. Then read one line for each of these modes, containing first the number of the mode, and then the momentum. The default is to have no momentum. See also the section describing how to perform a dynamic walk, Sec. 7.2.2.

.NATCON Use the natural connection when orthogonalizing the predicted molecular orbitals at the new geometry. By default the symmetric connection is used.

.NEWTON Use a strict Newton–Raphson step to update the geometry. This means that no trust region will be used.

.NO CENTRIFUGAL FORCES Do not include contributions from centrifugal forces when calculating vibrationally averaged geometries at a finite temperature.

.NOGRAD

```
    READ (LUCMD, *) NZEROG
    READ (LUCMD, *) (IZEROG(I), I = 1, NZEROG)
```

Set some gradient elements to zero. Read one more line specifying how many elements to zero, then one or more lines listing their sequence numbers.

.NOORTH The predicted molecular orbitals at the new geometry are *not* orthogonalized. Default is that the orbitals are orthogonalized with the symmetric connection. Orthogonalization can also be done with the natural connection [64]. See the keyword .NATCON.

.NOPRED No prediction of the energy of the wave function at the updated geometry.

.NORMAL Do the calculation of effective (vibrationally averaged) geometries in normal coordinates. This will restrict the calculation of the effective geometry to one isotopic species (by default the most abundant one).

.NUMERI Do a numerical differentiation, for instance when calculating Raman intensities or Raman optical activity, see Sections 8.4 and 11.4.

.PRINT

```
    READ (LUCMD,*) IPRWLK
```

Set the print level in the prediction of new geometry steps. Read one more line containing print level. Default value is the value of IPRDEF in the general input module.

`.RATLIM`

```
READ (LUCMD, *) RTRMIN, RTRGOD, REJMIN, REJMAX
```

Limits on ratios between predicted and observed energy change. Read one more line specifying four values (*). These are respectively the bad prediction ratio, good prediction ratio, low rejection ratio and high rejection ratio. Defaults are 0.4, 0.8, 0.1, and 1.9.

`.REJECT` Signals that the previous geometry step was rejected, and the trust region is reduced. This keyword is used in case of restarts to tell the program that when the program was stopped, the last geometry was in fact rejected.

`.REPS`

```
READ (LUCMD, *) NREPS
READ (LUCMD, *) (IDOREP(I), I = 1, NREPS)
```

Consider perturbations of selected symmetries only. Read one more line specifying how many symmetries, then one line listing the desired symmetries. Note that only those symmetries previously defined to be true with the keyword `.REPS` from the ABACUS input modules will be calculated. This keyword thus represents a subset of the `.REPS` of the general input module.

`.RESTART` Tells the program that this is a restarted geometry optimization and that information may therefore be available on the `DALTON.WLK` file.

`.REUSE` Use the property derivatives available on the file `DALTON.WLK` in a calculation of the harmonic contribution to the vibrational average. In this case, only a new force field will be calculated.

`.SCALE`

```
READ (LUCMD, *) NUMNUC
DO INUC = 1, NUMNUC
   READ (LUCMD, *) IATOM,(SCALCO(J,IATOM), J = 1, 3)
END DO
```

Scale the atomic coordinates. Read one more line specifying how many atoms to scale, then one line for each of these atoms (*) specifying the atom number and scale factors for all three Cartesian coordinates. Default is no scaling of the atomic coordinates.

`.TEMPERATURES`

```
READ (LUCMD, *) NTEMP
READ (LUCMD, *) (TEMP(ITMP), ITMP=1,NTEMP)
```

Read a set of temperatures for which the effective (rovibrationally averaged) geometries are to be calculated. Read one more line containing the number of different temperatures, and another line containing the list of temperatures.

.TOLERANCE

```
READ (LUCMD, *) TOLST
```

Threshold for convergence of the geometry optimization (on gradient norm). Read one more line specifying the threshold (*). Default is 10^{-5}.

.TRUST

```
READ (LUCMD, *) TRUSTR, TRUSTI, TRUSTD
```

Trust region information. Read one more line specifying three values (*): initial trust radius, factor by which radius can be incremented, and factor by which it can be decremented. Defaults are 0.5, 1.2 and 0.7, respectively; initial trust radius default is 0.3 if desired Hessian index is greater than zero. In dynamic walks the trust radius is by default put to 0.005, and in walks along an Intrinsic Reaction Coordinate (IRC) the default trust radius is 0.02. For dynamical walks the default increment and decrement factor is changed to 2.0 and 0.8 respectively.

.VIBAVE Request the calculation of the harmonic contribution to the vibrational average of a molecular property.

.ZERGRD

```
READ (LUCMD, *) ZERGRD
```

Threshold below which gradient elements are treated as zero. Read one more line specifying value (*). Default is 10^{-5}. This keyword is mainly used for judging which modes are symmetry breaking when using the keyword **.KEEPSY** as well as when deciding what step to take when starting a walk from a transition state.

21.2 Numerical differentiation : **∗∗NMDDRV**

This module can calculate any geometrical derivative of the energy using either as high analytical derivatives as possible, or using the specified level of analytical derivatives (assuming implemented for the choice of wave function) [113]. Also performs vibrational averaging over selected first- and second-order molecular properties.

.DISPLACEMENT

```
READ (LUCMD,*) DISPLC
```

Reads in the step lengths (in atomic units) that is to be used in the numerical differentiation scheme. Default is 1.0D-2.

.DORDR

    READ (LUCMD,*) NMORDR, NAORDR

Sets the numerical (NMORDR) and analytical (NAORDR) differentiating order for calculating force constants. Current implementation has an artificial boundary at the 5. numerical derivative (independent of the analytical differentiation order). Notice that if you would like to calculating 4.derivatives from analytical 2.derivatives the input would be 2 2, since you would like to get 2.order numerical derivatives from 2.order analytical derivatives.

.DRYRUN

    READ (LUCMD,*) NMREDU
    READ (LUCMD,*) (KDRYRN(II),II=1,NMREDU)

The numerical derivatives will not be calculated, and the program will just set up the required displacement. An optional number of redundant coordinate displacements can be specified, corresponding to translational and rotational degrees of freedom.

.HARMONIC FORCE FIELD

At the end of the calculation, perform a vibrational analysis using the calculated Hessian matrix.

.MANUAL

Dump each individual geometry input file to the `DALTON.OUT` file. Primarily of interest for debugging purposes.

.NORMAL Using this keyword the numerical differentiation will be carried out with respect to normal coordinates. The program will then do the necessary numerical differentiation to get the Hessian (and thus normal coordinates), before it will carry on calculating the higher-order force field requested with respect to the calculated normal coordinates.

.PRECALCULATED HESSIAN

Use a precalculated Hessian available on the `DALTON.HES` file when defining normal coordinates. Only active in combination with the keyword `.NORMAL`.

.PRINT

    READ (LUCMD,*) IDRPRI

Control the print level in the numerical derivative routines. Default is the same as the general print level `IPRUSR`.

.PROPAV

Indicates that a averaging over vibrational motions of a molecular property (including the energy) is to be performed. The input for what kind of a vibrational analysis is to be performed is specified in the `*PROPAV` module.

.PROPER
```
READ (LUCMD,*) NMRDRP, NARDRP
```
Sets the numerical (NMRDRP) and analytical (NARDRP) differentiating order for calculating geometrical derivatives of molecular properties. Currently, the possible presence of analytical property derivatives cannot be taken advantage of, and the default value of 0 for `NARDRP` should be used. Currently, dipole transition strengths and vibrationally averaged spin–spin coupling constants have been implemented, the former also for Coupled–Cluster wave functions. Notice that vibrationally averaged spin–spin coupling constants also can be calculated using the `.VIBANA` and `*VIBANA` keywords and input section.

.RESTRT Controls the restart procedure. If a calculation crashes during a force constant calculation, there will be a file in the work directory called RSTRT.FC. This will contain all of the information necessary to restart the calculation. If this file is available and the keyword is used, DALTON will attempt to restart the calculation.

.REUSE HESSIAN

If a (mixed) numerical Hessian has been calculated, it will be saved in the file `DALTON.HES` for possible future use.

.SDRTST

If analytical Hessian also has been requested and second-order numerical derivatives, a comparison of the numerical and analytical Hessian will be done. Primarily used for testing.

.SPECTRO INTERFACE

Write an interface file `DALTON.SPC` containing force fields of different orders suited for analysis with the SPECTRO program [?].

.SYMMET
```
READ (LUCMD,*) FCLASS
```

Assigns the molecular point group of the molecule (FCLASS). For instance for water FCLASS would be equal to "C2v". Main rotational axis needs to be set to the z-axis in the .mol file. Additional generating elements needs to be the x-axis for a C2 rotation, and the xy plane for a mirror plane. The current implementation only allows for symmetry use, when differentiating from energies.

.TEST N

Test if the normal coordinates used for calculated geometrical derivatives give rise to force fields with appropriate symmetries. Mainly for debugging purposes.

21.2.1   Vibrational averaging of molecular properties: *PROPAV

This module sets up overall numerical or mixed-numerical/analytical geometrical property derivative calculations, as well as performs selected post-analyses of the calculated energy and property derivatives in terms of zero-point vibrationally averaged properties.

.ANHA-P

Requests the calculation of the anharmonic contribution to a vibrationally averaged molecular property, requiring the use of the first-order perturbed vibrational wave function (requiring knowledge about parts of the cubic force field) and the first derivative of the molecular property. See also the keyword .HARM-P. Currently only implemented for spin–spin coupling constants.

.EFFECTIVE GEOMETRY

The effective geometry is calculated. This geometry corresponds to the zero-point vibrationally averaged geometry and is thus often referred to as the r_z geometry. For more information on the effective geometry, see Refs. [14, 15].

.HARM-P

Requests the calculation of the harmonic contribution to a vibrationally averaged molecular property, requiring the use of the unperturbed vibrational wave function and the second derivative of the molecular property. See also the keyword .ANHA-P. Currently only implemented for spin–spin coupling constants.

.MODE ANALYSIS

Request an analysis of the contribution of the different vibrational modes to the total zero-point vibrational corrections to a molecular property.

`.SPIN-SPIN COUPLINGS`

Requests the calculation of indirect spin–spin coupling constants. If combined with the keywords `.HARM-P` and `.ANHA-P`, the zero-point vibrational corrections to these coupling constants will be calculated.

21.2.2 Vibrational analysis: *VIBANA

This section is identical to the vibrational analysis module described in Chapter 25.1.21, but appears in the `**NMDDRV` module if the Hessian has been calculated numerically using the `.NMDDRV` keyword. We refer to Chapter 25.1.21 for a list of available keywords.

Chapter 22

Integral evaluation, HERMIT

22.1 General

HERMIT is the integral evaluation part of the code. In ordinary calculations there is no need to think about integral evaluation, as this will be automatically taken care of by the program. However, HERMIT has an extensive set of atomic one- and two-electron integrals, and some users may find it useful to generate explicit integrals using HERMIT. This is for instance necessary if the RESPONSE program (dynamic properties) is to be used, as described in Chapter 26. Disk usage may also be reduced by not calculating the supermatrix, and this is also controlled in the **INTEGRALS input section.

It is worth noticing that the two-electron part of HERMIT is actually two integral programs. TWOINT is the more general one and is invoked by default in sequential calculations; ERI (Electron Repulsion Integrals) is a highly vectorized two-electron integral code with orientation towards integral distributions. ERI is invoked by default in integral-direct coupled cluster calculations and may in other cases be invoked by specifying the .RUNERI keyword in the **DALTON INPUT input section, which ensures that ERI rather than TWOINT is called whenever ERI has the required functionality. DALTON will, however, automatically revert to TWOINT for any two-electron integral not available in ERI: ERI cannot be used in parallel calculations and it contains only first-derivatives (for the Hartree–Fock gradient), thus it cannot be used to calculate Hessians or to calculate MCSCF gradients.

Input to integral evaluation is indicated by the keyword **INTEGRALS, and the section may be ended with *END OF or any keyword starting with two stars (like *e.g.* **WAVE FUNCTIONS). The intermediate input is divided into two sections: one general input section describing what molecular integrals are to be evaluated, and then a set of modules controlling the different parts of the calculation of atomic integrals and the (possible) formation of a supermatrix as defined in for instance Ref. [114].

22.2 **INTEGRALS directives

The following directives may be included in the input to the integral evaluation. They are organized according to the program section names in which they can appear.

22.2.1 End of input: *END OF

The last directive of the **INTEGRALS input may be *END OF.

22.2.2 General: **INTEGRALS

General-purpose directives are given in the **INTEGRALS section. This mainly includes requests for different atomic integrals, as well as some directives affecting the outcome of such an integral evaluation. Note that although not explicitly stated, *none of the test options work with symmetry.*

For all atomic integrals, the proper expression for the integral is given, together with the labels written on the file AOPROPER, for reference in later stages of a DALTON calculation (like for instance in during the evaluation of dynamic response properties, or for non-DALTON programs).

We also note that as long as any single atomic property integral is requested in this module, the overlap integrals will also be calculated. Note also, that unless the Hückel starting guess is turned off, this overlap matrix will not only be calculated for the requested basis set, but also for a "ghost" ano-4 basis set appended to the original set in order to do the Hückel starting guess.

.1ELPOT One-electron potential energy integrals.

$$\text{Integral: } \sum_K \left\langle \chi_\mu \left| \frac{Z_K}{r_K} \right| \chi_\nu \right\rangle$$

Property label: POTENERG

.AD2DAR

    READ (LUCMD,*) DARFAC

$$\text{Integral: } \frac{\alpha^2}{4} \left\langle \chi_\mu \chi_\nu \left| \delta\left(\mathbf{r}_{12}\right) \right| \chi_\rho \chi_\sigma \right\rangle$$

Add two-electron Darwin integrals to the standard electron-repulsion integrals with a perturbation factor DARFAC.

.ANGLON Contribution to the one-electron contribution of the magnetic moment using London orbitals arising from the differentiation of London-orbital transformed Hamiltonian, see Ref. [115].

Integral: $\langle \chi_\mu \left| \mathbf{L}_N \right| \chi_\nu \rangle$

Property labels: `XANGLON` , `YANGLON` , `ZANGLON`

`.ANGMOM` Angular momentum around the molecular origin. This can be adjusted by changing the gauge origin through the use of the `.GAUGEO` keyword.

Integral: $\langle \chi_\mu \left| \mathbf{L}_O \right| \chi_\nu \rangle$

Property labels: `XANGMOM` , `YANGMOM` , `ZANGMOM`

`.AUXBAS` An auxiliary basis is used. Basis sets must be identified as orbital basis and auxiliary basis in the `MOLECULE.INP` file (line 5).

`.CARMOM`

    `READ (LUCMD,*) IORCAR`

Cartesian multipole integrals to order `IORCAR`. Read one more line specifying order. See also the keyword `.SPHMOM`.

Integral: $\left\langle \chi_\mu \left| x^i y^j z^k \right| \chi_\nu \right\rangle$

Property labels: `CMiijjkk`

where $ii + jj + kk =$ `IORDER`, and where `ii` = (i/10)*10+mod(i,10).

`.CM-1`

    `READ (LUCMD, '(A7)') FIELD1`

First derivative of the electric dipole operator with respect to an external magnetic field due to differentiation of the London phase factors, see Ref. [92]. Read one more line giving the direction of the electric field (A7). These include `X-FIELD`, `Y-FIELD`, and `Z-FIELD`.

Integral: $Q_{MN} \langle \chi_\mu \left| \mathbf{r}D \right| \chi_\nu \rangle$

Property labels: `D-CM1 X` , `D-CM1 Y` , `D-CM1 Z`

where D is the direction of the applied electric field as specified in the input.

`.CM-2`

    `READ (LUCMD, '(A7)') FIELD2`

Second derivative electric dipole operator with respect to an external magnetic field due to differentiation of the London phase factors, see Ref. [92]. Read one more line giving the direction of the electric field (A7). These include `X-FIELD`, `Y-FIELD`, and `Z-FIELD`.

Integral: $Q_{MN} \left\langle \chi_\mu \left| \mathbf{rr}^T D \right| \chi_\nu \right\rangle Q_{MN}$

Property labels: `D-CM2 XX, D-CM2 XY, D-CM2 XZ, D-CM2 YY, D-CM2 YZ, D-CM2 ZZ`

where D is the direction of the applied electric field as specified in the input.

`.DARWIN` One-electron Darwin integrals [116].

Integral: $\frac{\pi\alpha^2}{2} \left\langle \chi_\mu \left| \delta\left(\mathbf{r}\right) \right| \chi_\nu \right\rangle$

Property label: `DARWIN`

`.DCCR12` Only required for interfaces to other implementations of the R12 approach. Obsolete, do not use.

`.DEROVL` Geometrical first derivatives of overlap integrals.

Integral: $\frac{\partial}{\partial \mathbf{R}_K} \left\langle \chi_\mu \mid \chi_\nu \right\rangle$

Property labels: `1DOVLxyz`

where xyz is the symmetry adapted nuclear coordinate.

`.DERHAM` Geometrical first derivatives of the one-electron Hamiltonian matrix.

Integral: $\frac{\partial}{\partial \mathbf{R}_K} \left\langle \chi_\mu \left| \sum_K \frac{Z_K}{r_K} - \frac{1}{2}\nabla^2 \right| \chi_\nu \right\rangle$

Property labels: `1DHAMxyz`

where xyz is the symmetry adapted nuclear coordinate.

`.DIASUS` Diamagnetic magnetizability integrals, as calculated with London atomic orbitals, see Ref. [115]. It is calculated as the sum of the three contributions `DSUSLH`, `DSUSLL`, and `DSUSNL`.

Integral: $\frac{1}{4}\left[\left\langle \chi_\mu \left| r_N^2 I - \mathbf{r}_N \mathbf{r}_N^T \right| \chi_\nu \right\rangle + \overline{Q_{MN}\left\langle \chi_\mu \left| \mathbf{r}\mathbf{L}_N^T \right| \chi_\nu \right\rangle} + Q_{MN}\left\langle \chi_\mu \left| \mathbf{rr}^T h \right| \chi_\nu \right\rangle Q_{MN} \right]$

Property label: `XXdh/dB2, XYdh/dB2, XZdh/dB2, YYdh/dB2, YZdh/dB2, ZZdh/dB2`

`.DIPGRA` Calculate dipole gradient integrals, that is, the geometrical first derivatives of the dipole length integrals.

Integral: $\frac{\partial}{\partial \mathbf{R}_K} \left\langle \chi_\mu \left| \mathbf{r} \right| \chi_\nu \right\rangle$

Property labels: `abcDPG d`

where abc is the symmetry adapted nuclear coordinate, and d the direction (x/y/z) of the dipole moment.

.DIPLEN Dipole length integrals.

> Integral: $\langle \chi_\mu \left| \mathbf{r} \right| \chi_\nu \rangle$
>
> Property labels: `XDIPLEN` , `YDIPLEN` , `ZDIPLEN`

.DIPORG

> `READ (LUCMD, *) (DIPORG(I), I = 1, 3)`

Specify the dipole origin to be used in the calculation. Read one more line containing the three Cartesian components (*). Default is (0,0,0).

.DIPVEL Dipole velocity integrals.

> Integral: $\langle \chi_\mu \left| \nabla \right| \chi_\nu \rangle$
>
> Property label: `XDIPVEL` , `YDIPVEL` , `ZDIPVEL`

.DNS-KE Kinetic-energy correction to the diamagnetic contribution to nuclear shielding constants with a common gauge origin, see Ref. [**?**].

> Integral: $\frac{3}{4} \left\langle \chi_\mu \left| \left[\nabla^2, \frac{\mathbf{r}_O^T \mathbf{r}_K - \mathbf{r}_O \mathbf{r}_K^T}{r_K^3} \right]_+ \right| \chi_\nu \right\rangle$
>
> Property label: `abcNSKEd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the magnetic field. O is the gauge origin.

.DPTOVL DPT (Direct Perturbation Theory) integrals: Small-component one-electron overlap integrals.

> Integral: $\left\langle \chi_\mu \left| \frac{\partial^2}{\partial \mathbf{r}^2} \right| \chi_\nu \right\rangle$
>
> Property labels: `dd/dxdx`, `dd/dxdy`, `dd/dxdz`, `dd/dydy`, `dd/dydz`, `dd/dzdz`

.DPTPOT DPT (Direct Perturbation Theory) integrals: Small-component one-electron potential energy integrals.

> Integral: $\left\langle \chi_\mu \left| \frac{\partial}{\partial \mathbf{r}} \frac{1}{\mathbf{R}_K} \frac{\partial}{\partial \mathbf{r}} \right| \chi_\nu \right\rangle$
>
> Property labels: `DERXXPVP`, `DERXY+YX`, `DERXZ+ZX`, `DERYY`, `DERYZ+ZY`, `DERZZ`

.DPTPXP DPT (Direct Perturbation Theory) integrals: Small-component dipole length integrals for Direct Perturbation Theory.

> Integral: $\frac{1}{4} \langle \nabla \chi_\mu \left| \mathbf{r} \right| \nabla \chi_\nu \rangle$

Property labels: `PXPDIPOL`, `PYPDIPOL`, `PZPDIPOL`.

.DSO Diamagnetic spin-orbit integrals. These are calculated using Gaussian quadrature as described in Ref. [117]. The number of quadrature point is controlled by the keyword `.POINTS`.

Integral: $\left\langle \chi_\mu \left| \frac{\mathbf{r}_K^T \mathbf{r}_L I - \mathbf{r}_K \mathbf{r}_L^T}{r_K^3 r_L^3} \right| \chi_\nu \right\rangle$

Property labels: `DSO abcd` where `ab` is the symmetry coordinate of a given component for the symmetry-adapted nucleus K, and `cd` is in a similar fashion the symmetry coordinate for the symmetry-adapted nucleus L.

.DSO-KE Kinetic energy correction to the diamagnetic spin-orbit integrals. These are calculated using Gaussian quadrature as described in Ref. [117]. The number of quadrature point is controlled by the keyword `.POINTS`. Please note that this integral has not been extensively tested, and the use of this integral is at the risk of the user.

Integral: $\left\langle \chi_\mu \left| \left[\nabla^2, \frac{\mathbf{r}_K^T \mathbf{r}_L I - \mathbf{r}_K \mathbf{r}_L^T}{r_K^3 r_L^3} \right]_+ \right| \chi_\nu \right\rangle$

Property labels: `DSOKabcd` where `ab` is the symmetry coordinate of a given component for the symmetry-adapted nucleus K, and `cd` is in a similar fashion the symmetry coordinate for the symmetry-adapted nucleus L.

.DSUSLH The contribution to diamagnetic magnetizability integrals from the differentiation of the London orbital phase-factors, see Ref. [115].

Integral:$\frac{1}{4} Q_{MN} \left\langle \chi_\mu \left| \mathbf{r} \mathbf{r}^T h \right| \chi_\nu \right\rangle Q_{MN}$

Property labels: `XXDSUSLH, XYDSUSLH, XZDSUSLH, YYDSUSLH, YZDSUSLH, ZZDSUSLH`

.DSUSLL The contribution to the diamagnetic magnetizability integrals from mixed differentiation on the Hamiltonian and the London orbital phase factors, see Ref. [115].

Integral: $\frac{1}{4} \overline{Q_{MN} \left\langle \chi_\mu \left| \mathbf{r} \mathbf{L}_N^T \right| \chi_\nu \right\rangle}$

Property labels: `XXDSUSLL, XYDSUSLL, XZDSUSLL, YYDSUSLL, YZDSUSLL, ZZDSUSLL`

.DSUSNL The contribution to the diamagnetic magnetizability integrals using London orbitals but with contributions from the differentiation of the Hamiltonian only, see Ref. [115].

Integral: $\frac{1}{4} \left\langle \chi_\mu \left| r_N^2 I - \mathbf{r}_N \mathbf{r}_N^T \right| \chi_\nu \right\rangle$

Property labels: `XXDSUSNL, XYDSUSNL, XZDSUSNL, YYDSUSNL, YZDSUSNL, ZZDSUSNL`

.DSUTST Test of the diamagnetic magnetizability integrals with London atomic orbitals. Mainly for debugging purposes.

.EFGCAR Cartesian electric field gradient integrals.

Integral: $\frac{1}{3} \left\langle \chi_\mu \left| \frac{3\mathbf{r}_K \mathbf{r}_K^T - \mathbf{r}_K^T \mathbf{r}_K I}{r_K^5} \right| \chi_\nu \right\rangle$

Property labels: `xyEFGabc`, where `x` and `y` are the Cartesian directions, `abc` the number of the symmetry independent center, and `c` that centers c'th symmetry-generated atom.

.EFGSPH Spherical electric field gradient integrals. Obtained by transforming the Cartesian electric-field gradient integrals (see **.EFGCAR**) to spherical basis.

.ELGDIA Diamagnetic one-electron spin-orbit integrals without London orbitals.

Integral:

Property labels: `D1-SO XX`, `D1-SO XY`, `D1-SO XZ`, `D1-SO YX`, `D1-SO YY`, `D1-SO YZ`, `D1-SO ZX`, `D1-SO ZY`, `D1-SO ZZ`

.ELGDIL Diamagnetic one-electron spin-orbit integrals with London orbitals.

Integral:

Property labels: `D1-SOLXX`, `D1-SOLXY`, `D1-SOLXZ`, `D1-SOLYX`, `D1-SOLYY`, `D1-SOLYZ`, `D1-SOLZX`, `D1-SOLZY`, `D1-SOLZZ`

.EXPIKR

```
READ (LUCMD, *) (EXPKR(I), I = 1, 3)
```

Cosine and sine integrals. Read one more line containing the wave numbers in the three Cartesian directions. The center of expansion is always (0,0,0).

Integral:

Property labels: `COS KX/K`, `COS KY/K`, `COS KZ/K`, `SIN KX/K`, `SIN KY/K`, `SIN KZ/K`.

.FC Fermi-contact integrals, see Ref. [61].

Integral: $\frac{4\pi g_e}{3} \left\langle \chi_\mu \left| \delta\left(\mathbf{r}_K\right) \right| \chi_\nu \right\rangle$

Property labels: `FC NAMab`, where `NAM` is the three first letters in the name of this atom, as given in the `MOLECULE.INP` file, and `ab` is the number of the symmetry-adapted nucleus.

.FC-KE Kinetic energy correction to Fermi-contact integrals, see Ref. [**?**].

> Integral: $\frac{2\pi g_e}{3} \left\langle \chi_\mu \left| \left[\nabla^2, \delta\left(\mathbf{r}_K\right) \right]_+ \right| \chi_\nu \right\rangle$
>
> Property labels: `FCKEnacd`, where `na` is the two first letters in the name of this atom, as given in the `MOLECULE.INP` file, and `cd` is the number of the symmetry-adapted nucleus.

.FINDPT

> `READ (LUCMD, *) DPTFAC`

A direct relativistic perturbation is added to the Hamiltonian and metric with the perturbation parameter DPTFAC, where the actually applied perturbation is DPTFAC*α_{fs}^2.

.GAUGEO

> `READ (LUCMD, *) (GAGORG(I), I = 1, 3)`

Specify the gauge origin to be used in the calculation. Read one more line containing the three Cartesian components (*). Default is (0,0,0).

.HBDO Symmetric combination of half-differentiated overap matrix with respect to an external magnetic field perturbation when London orbitals are used.

> Integral: $-\frac{1}{4}\left(Q_{MO} + Q_{NO}\right) \left\langle \chi_\mu \left| \mathbf{r} \right| \chi_\nu \right\rangle$
>
> Property labels: `HBDO X` , `HBDO Y` , `HBDO Z` .

.HDO Symmetrized, half-differentiated overlap integrals with respect to geometric distortions, see Ref. [118]. Differentiation on the ket-vector.

> Integral: $\left\langle \frac{\partial \chi_\mu}{\partial R_{ab}} \mid \chi_\nu \right\rangle - \left\langle \chi_\mu \mid \frac{\partial \chi_\nu}{\partial R_{ab}} \right\rangle$
>
> Property label: `HDO abc` , where `abc` is the number of the symmetry-adapted coordinate being differentiated.

.HDOBR Geometric half-differentiated overlap matrix differentiated once more on the ket-vector with respect to an external magnetic field, see Ref. [78].

> Integral: $-\frac{1}{2} Q_{NO} \left\langle \frac{\partial \chi_\mu}{\partial R_K} \left| \mathbf{r} \right| \chi_\nu \right\rangle$
>
> Property labels: `abcHBD d`, where `abc` is the symmetry coordinate of the nuclear coordinate being differentiations, and `d` is the coordinate of the external magnetic field.

.HDOBRT Test the calculation of the `.HDOBR` integral. Mainly for debugging purposes.

.INPTES Test the correctness of the `**INTEGRALS`-input. Mainly for debugging purposes, but also a good option to check if the MOLECULE input has been typed in correctly.

.KINENE Kinetic energy integrals. Note however, that the kinetic energy integrals used in the wave function optimization is generated in the `*ONEINT` section.

> Integral: $\frac{1}{2}\left\langle \chi_\mu \left| \nabla^2 \right| \chi_\nu \right\rangle$
>
> Property label: `KINENERG`.

.LONMOM Contribution to the London magnetic moment from the differentiation with respect to magnetic field on the London orbital phase factors, see Ref. [115].

> Integral: $\frac{1}{4}Q_{MN}\left\langle \chi_\mu \left| \mathbf{r}h \right| \chi_\nu \right\rangle$
>
> Property labels: `XLONMOM` , `YLONMOM` , `ZLONMOM` .

.MAGMOM One-electron contribution to the magnetic moment around the nuclei to which the atomic orbitals are attached. This is the London atomic orbital magnetic moment as defined in Eq. (35) of Ref. [56]. The integral is calculated as the sum of `.LONMOM` and `.ANGLON`.

> Integral: $\left\langle \chi_\mu \left| \mathbf{L}_N + \frac{1}{4}Q_{MN}\mathbf{r}h \right| \chi_\nu \right\rangle$
>
> Property label: `dh/dBX` , `dh/dBY` , `dh/dBZ` .

.MASSVE Mass-velocity integrals.

> Integral: $\frac{\alpha^2}{8}\left\langle \chi_\mu \left| \nabla^2 \cdot \nabla^2 \right| \chi_\nu \right\rangle$
>
> Property label: `MASSVELO`.

.MGMO2T Test of two-electron integral contribution to magnetic moment.

.MGMOMT Test the calculation of the `.MAGMOM` integrals.

.MGMTHR

> `READ (LUCMD, *) PRTHRS`
>
> Set the threshold for which two-electron integrals should be tested with the keyword `.MGMO2T`. Default is 10^{-10}.

.MNF-SO Calculates the atomic mean-field spin-orbit integrals as described in Ref. [103]. As the calculation of these integrals require a proper description of the atomic states, reliable results can only be expected for generally contracted basis sets such as the ANO sets, and in some cases also the correlation-consistent basis sets ((aug-)cc-p(C)VXZ).

`.NELFLD` Nuclear electric field integrals.

Integral: $\left\langle \chi_\mu \left| \frac{\mathbf{r}_K}{r_K^3} \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.

Property labels: `NEF abc` , where `abc` is the number of the symmetry-adapted nuclear coordinate.

`.NO HAM` Do not calculate ordinary one- and two-electron Hamiltonian integrals.

`.NO2SO` Do not calculate two-electron contribution to spin–orbit integrals.

`.NOPICH` Do not add direct perturbation theory correction to Hamiltonian integral, see keyword `.FINDPT`.

`.NOSUP` Do not calculate the supermatrix integral file. This may be required in order to reduce the amount of disc space used in the calculation (to approximately one-third before entering the evaluation of molecular properties). Note however, that this will increase the time used for the evaluation of the wave function significantly in ordinary Hartree–Fock runs. It is default for direct and parallel calculations.

`.NOTV12` Obsolete keyword, do not use.

`.NOTWO` Only calculate the one-electron part of the Hamiltonian integrals. It is default for direct and parallel calculations.

`.NPOTST` Test of the nuclear potential integrals calculated with the keyword `.NUCPOT`. Mainly for debugging purposes.

`.NSLTST` Test of the integrals calculated with the keyword `.NSTLON`. Mainly for debugging purposes.

`.NSNLTS` Test of the integrals calculated with the keyword `.NSTNOL`. Mainly for debugging purposes.

`.NST` Calculate the one-electron contribution to the diamagnetic nuclear shielding tensor integrals using London atomic orbitals, see Ref. [115]. It is calculated as the sum of `NSTLON` and `NSTNOL`.

Integral: $\frac{1}{2} \left\langle \chi_\mu \left| \frac{\mathbf{r}_N^T \mathbf{r}_K - \mathbf{r}_N \mathbf{r}_K^T}{r_K^3} + Q_{MN} \frac{\mathbf{r}_N^T \mathbf{l}_K}{r_K^3} \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.

Property label: `abcNST d`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the magnetic field.

.NSTCGO Calculate the diamagnetic nuclear shielding tensor integrals without using London atomic orbitals. Note that the gauge origin is controlled by the keyword **.GAUGEO**.

Integral: $\frac{1}{2}\left\langle\chi_\mu\left|\frac{\mathbf{r}_O^T\mathbf{r}_K-\mathbf{r}_O\mathbf{r}_K^T}{r_K^3}\right|\chi_\nu\right\rangle$ where K is the nucleus of interest.

Property label: `abcNSCOd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the magnetic field. O is the gauge origin.

.NSTLON Calculate the contribution to the London orbital nuclear shielding tensor from the differentiation of the London orbital phase-factors, see Ref. [115].

Integral: $\frac{1}{2}Q_{MN}\left\langle\chi_\mu\left|\frac{\mathbf{r}_N^T\mathbf{l}_K}{r_K^3}\right|\chi_\nu\right\rangle$ where K is the nucleus of interest.

Property labels: `abcNSLOd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the magnetic field.

.NSTNOL Calculate the contribution to the nuclear shielding tensor when using London atomic orbitals from the differentiation of the Hamiltonian alone, see Ref. [115].

Integral: $\frac{1}{2}\left\langle\chi_\mu\left|\frac{\mathbf{r}_N^T\mathbf{r}_K-\mathbf{r}_N\mathbf{r}_K^T}{r_K^3}\right|\chi_\nu\right\rangle$ where K is the nucleus of interest.

Property label: `abcNSNLd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the magnetic field.

.NSTTST Test the calculation of the one-electron diamagnetic nuclear shielding tensor using London atomic orbitals.

.NUCMOD

```
READ (LUCMD, *) INUC
```

Choose nuclear model. A 1 corresponds to a point nucleus (which is the default), and 2 corresponds to a Gaussian distribution model.

.NUCPOT Calculate the nuclear potential energy. Currently this keyword can only be used in calculations not employing symmetry.

Integral: $\left\langle\chi_\mu\left|\frac{Z_K}{r_K}\right|\chi_\nu\right\rangle$ where K is the nucleus of interest.

Property labels: `POT.E ab`, where `ab` are the two first letters in the name of this nucleus. Thus note that in order to distinguish between integrals, the first two letters in an atom's name must be unique.

.OCTGRA Calculate octupole gradient integrals, that is, the geometrical first derivatives of the third moment integrals (`.THIRDM`).

> Integral: $\frac{\partial}{\partial \mathbf{R}_K} \left\langle \chi_\mu \left| \mathbf{r}^3 \right| \chi_\nu \right\rangle$
>
> Property labels: `abODGcde`

where `ab` is the symmetry adapted nuclear coordinate, and `cde` the component (x/y/z) of the third moment tensor. Currently, this integral does not work with symmetry.

.OZ-KE Calculates the kinetic energy correction to the orbital Zeeman operator, see Ref. [**?**].

> Integral: $\left\langle \chi_\mu \left| \left[\nabla^2, \mathbf{l}_O \right]_+ \right| \chi_\nu \right\rangle$
>
> Property labels: `XOZKE` , `YOZKE` , `ZOZKE` .

.PHASEO
>   READ (LUCMD, *) (ORIGIN(I), I = 1, 3)

Set the origin appearing in the London atomic orbital phase-factors. Read one more line containing the Cartesian components of this origin (*). Default is (0,0,0).

.POINTS
>   READ (LUCMD,*) NPQUAD

Read the number of quadrature points to be used in the evaluation of the diamagnetic spin-orbit integrals, as requested by the keyword `.DSO`. Read one more line containing the number of quadrature points. Default is 40.

.PRINT
>   READ (LUCMD,*) IPRDEF

Set default print level during the integral evaluation. Read one more line containing print level. Default is the value of `IPRDEF` from the general input module for DALTON.

.PROPRI Print all one-electron property integrals requested.

.PSO Paramagnetic spin-orbit integrals, see Ref. [61].

> Integral: $\left\langle \chi_\mu \left| \frac{\mathbf{l}_K}{r_K^3} \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.
>
> Property label: `PSO abc` , where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate.

.PSO-KE Kinetic energy correction to the paramagnetic spin-orbit integrals, see Ref. [**?**].

Integral: $\left\langle \chi_\mu \left| \left[\nabla^2, \frac{\mathbf{l}_K}{r_K^3} \right]_+ \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.

Property label: `PSOKEabc`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate.

.PSO-OZ Orbital-Zeeman correction to the paramagnetic spin-orbit integrals, see Ref. [**?**].

Integral: $\left\langle \chi_\mu \left| \left[\mathbf{l}_O, \frac{\mathbf{l}_K}{r_K^3} \right]_+ \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.

Property label: `abcPSOZd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and d is the direction (x/y/z) of the external magnetic field (corresponding to the component of the orbital Zeeman operator).

.PVP Calculate the pVp integrals that appear in the Douglas–Kroll–Heß transformation [119].

Integral: $\left\langle \chi_\mu \left| \nabla \left(\sum_K \frac{Z_K}{r_i K} \right) \nabla \right| \chi_\nu \right\rangle$

Property labels: `pVpINTEG`.

.PVIOLA Parity-violating electroweak interaction.

Integral:

Property labels: `PVIOLA X`, `PVIOLA Y`, `PVIOLA Z`.

.QDBINT

```
READ (LUCMD,'(A7)') FIELD3
```

London orbital corrections arising from the second-moment of charge operator in finite-perturbation calculations involving an external electric field gradient. Possible values for the perturbation (FIELD3) may be XX/XY/XZ/YY/YZ/ZZ-FGRD.

Integral:

Property labels: `ab-QDB X`, `ab-QDB Y`, `ab-QDB Z`, where ab is the component of the electric field gradient operator read in the variable FIELD3.

.QDBTST Test of the `.QDBINT` integrals, mainly for debugging purposes.

.QUADRU Quadrupole moment integrals. For traceless quadrupole moment integrals as defined by Buckingham [43], see the keyword `.THETA`.

Integral: $\frac{1}{4} \left\langle \chi_\mu \left| r_O^2 I_3 - \mathbf{r}_O \mathbf{r}_O^T \right| \chi_\nu \right\rangle$

Property label: `XXQUADRU`, `XYQUADRU`, `XZQUADRU`, `YYQUADRU`, `YZQUADRU`, `ZZQUADRU`

.QUAGRA

Calculate quadrupole gradient integrals, that is, the geometrical first derivatives of the second moment integrals (*i.e.* `.SECMOM`, note: it is NOT the gradient of the `.QUADRU` integrals).

Integral: $\frac{\partial}{\partial \mathbf{R}_K} \left\langle \chi_\mu \left| \mathbf{r}\mathbf{r}^T \right| \chi_\nu \right\rangle$

Property labels: `abcQDGde`

where `abc` is the symmetry adapted nuclear coordinate, and `de` the component (xx/xy/xz/yy/yz/zz) of the second moment tensor. Currently symmetry can not be used with these integrals.

.QUASUM Calculate all atomic integrals as square matrices, irrespective of their inherent Hermiticity or anti-Hermiticity.

.RANGMO Calculate the diamagnetic magnetizability integrals using the CTOCD-DZ method, see Ref. [58, 59]. The gauge origin is, as default, in the center of mass.

Integral: $\left\langle \chi_\mu \left| \mathbf{r}_O \mathbf{L}_N^T \right| \chi_\nu \right\rangle$

Property label: `XXRANG`, `XYRANG`, `XZRANG`, `YXRANG`, `YYRANG`, `YZRANG`, `ZXRANG`, `ZYRANG`, `ZZRANG`

.R12 Perform integral evaluation as required by the R12 method. One-electron integrals for the R12 method (Cartesian multipole integrals up to order 2) are precomputed and stored on the file `AOPROPER`. Two-electron integrals are computed in direct mode.

.R12EXP

`READ (LUCMD,*) GAMMAC`

Same as `.R12` but with Gaussian-damped linear r_{12} terms of the form $r_{12} \exp -\gamma r_{12}^2$. The value of γ is read from the input line.

.R12INT Calculation of two-electron integrals over r12.

.ROTSTR Rotational strength integrals in the mixed representation [120].

Integral: $\left\langle \chi_\mu \left| \nabla \, \mathbf{r}^T + \mathbf{r}\nabla^T \right| \chi_\nu \right\rangle$

Property labels : `XXROTSTR`, `XYROTSTR`, `XZROTSTR`, `YYROTSTR`, `YZROTSTR`, `ZZROTSTR`.

.RPSO Calculate the diamagnetic nuclear shielding tensor integrals using the CTOCD-DZ method, see Ref. [58, 59, 60]. The gauge origin is, as default, at the center of mass. Setting the gauge origin somewhere else will give wrong results in calculations using symmetry.

Integral: $\left\langle \chi_\mu \left| \frac{\mathbf{r}_O^T \mathbf{l}_K}{r_K^3} \right| \chi_\nu \right\rangle$ where K is the nucleus of interest.

Property label: `abcRPSOd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate and `d` refers to the x, y, z component of the magnetic field

.S1MAG Calculate the first derivative overlap matrix with respect to an external magnetic field by differentiation of the London phase factors, see Ref. [115].

Integral: $\frac{1}{2} Q_{MN} \langle \chi_\mu | \mathbf{r} | \chi_\nu \rangle$

Property labels: `dS/dBX  `, `dS/dBY  `, `dS/dBZ`

.S1MAGL Calculate the first magnetic half-differentiated overlap matrix with respect to an external magnetic field as needed with the natural connection, see Ref. [57]. Differentiated on the bra-vector.

Integral: $\frac{1}{2} Q_{MO} \langle \chi_\mu | \mathbf{r} | \chi_\nu \rangle$

Property label: `d<S|/dBX, d<S|/dBY, d<S|/dBZ`

.S1MAGR Calculate the first magnetic half-differentiated overlap matrix with respect to an external magnetic field as needed with the natural connection, see Ref. [57]. Differentiated on the ket-vector.

Integral: $\frac{1}{2} Q_{ON} \langle \chi_\mu | \mathbf{r} | \chi_\nu \rangle$

Property labels: `d|S>/dBX, d|S>/dBZ, d|S>/dBZ`

.S1MAGT Test the integrals calculated with the keyword **.S1MAG**. Mainly for debugging purposes.

.S1MLT Test the integrals calculated with the keyword **.S1MAGL**. Mainly for debugging purposes.

.S1MRT Test the integrals calculated with the keyword **.S1MAGR**. Mainly for debugging purposes.

.S2MAG Calculate the second derivative of the overlap matrix with respect to an external magnetic field by differentiation of the London phase factors, see Ref. [115].

Integral: $\frac{1}{4} Q_{MN} \left\langle \chi_\mu \left| \mathbf{rr}^T \right| \chi_\nu \right\rangle Q_{MN}$

Property labels: `dS/dB2XX`, `dS/dB2XY`, `dS/dB2XZ`, `dS/dB2YY`, `dS/dB2YZ`, `dS/dB2ZZ`

.S2MAGT Test the integrals calculated with the keyword **.S2MAG**. Mainly for debugging purposes.

.SD Spin-dipole integrals, see Ref. [61].

Integral: $\frac{g_e}{2} \left\langle \chi_\mu \left| \frac{3\mathbf{r}_K \mathbf{r}_K^T - r_K^2}{r_K^5} \right| \chi_\nu \right\rangle$

Property label: `SD abc d`, where `abc` is the number of the first symmetry-adapted coordinate (corresponding to symmetry-adapted nuclear magnetic moments) and `d` is the x, y, or z component of the magnetic moment with respect to spin coordinates.

.SD+FC Calculate the sum of the spin-dipole and Fermi-contact integrals.

Integral: $\frac{g_e}{2} \left\langle \chi_\mu \left| \frac{3\mathbf{r}_K \mathbf{r}_K^T - r_K^2}{r_K^5} \right| \chi_\nu \right\rangle + \frac{4\pi g_e}{3} \left\langle \chi_\mu \left| \delta \left(\mathbf{r}_K \right) \right| \chi_\nu \right\rangle$

Property label: `SDCabc d`, where `abc` is the number of the first symmetry-adapted coordinate (corresponding to symmetry-adapted nuclear magnetic moments) and `d` is the x, y, or z component of the magnetic moment with respect to spin coordinates.

.SD-KE Kinetic energy correction to spin–dipole integrals, see Ref. [**?**].

Integral: $\frac{g_e}{4} \left\langle \chi_\mu \left| \left[\nabla^2, \frac{3\mathbf{r}_K \mathbf{r}_K^T - r_K^2}{r_K^5} \right]_+ \right| \chi_\nu \right\rangle$

Property label: `SDKEab c`, where `ab` is the number of the first symmetry-adapted coordinate (corresponding to symmetry-adapted nuclear magnetic moments) and `c` is the x, y, or z component of the magnetic moment with respect to spin coordinates.

.SECMOM Second-moment integrals.

Integral: $\left\langle \chi_\mu \left| \mathbf{rr}^T \right| \chi_\nu \right\rangle$

Property labels: `XXSECMOM`, `XYSECMOM`, `XZSECMOM`, `YYSECMOM`, `YZSECMOM`, `ZZSECMOM`

.SELECT
```
READ (LUCMD, *) NPATOM
READ (LUCMD, *) (IPATOM(I), I = 1, NPATOM
```

Select which atoms for which a given atomic integral is to be calculated. This applies mainly to property integrals for which there exist a set of integrals for each nucleus. Read one more line containing the number of atoms selected, and then another line containing the numbers of the atoms selected. Most useful when calculating diamagnetic spin-orbit integrals, as this is a rather time-consuming calculation. The numbering is of symmetry-independent nuclei.

.SOFIEL External magnetic-field dependence of the spin–orbit operator integrals [121].

Integral: $\frac{1}{2} \sum_K Z_K \left\langle \chi_\mu \left| \frac{\mathbf{r}_O^T \mathbf{r}_K - \mathbf{r}_O \mathbf{r}_K^T}{r_K^3} \right| \chi_\nu \right\rangle$

Property labels: `SOMF  XX`, `SOMF  XY`, `SOMF XZ`, `SOMF  YX`, `SOMF  YY`, `SOMF  YZ`, `SOMF ZX`, `SOMF  ZY`, `SOMF  ZZ`.

.SOMAGM Nuclear magnetic moment dependence of the spin–orbit operator integrals [122].

Integral: $\sum_K Z_K \left\langle \chi_\mu \left| \frac{\mathbf{r}_K^T \mathbf{r}_L I - \mathbf{r}_K \mathbf{r}_L^T}{r_K^3 r_L^3} \right| \chi_\nu \right\rangle$

Property label: `abcSOMMd`, where `abc` is the number of the symmetry-adapted nuclear magnetic moment coordinate, and `d` refers to the x, y, or z component of the spin–orbit operator.

.SORT I Requests that the two-electron integrals should be sorted for later use in SIRIUS. See also keywords `.PRESORT` in the `**DALTON` and `*TRANSFORMATION` input sections.

.SOTEST Test the calculation of spin-orbit integrals as requested by the keyword `.SPIN-O`.

.SPHMOM

> `READ (LUCMD,*) IORSPH`

Spherical multipole integrals to order `IORSPH`. Read one more line specifying order. See also the keyword `.CARMOM`.

Property label: `CMiijjkk`

where $i + j + k =$`IORDER`, and where `ii` = (i/10)*10+mod(i,10).

.SPIN-O Spatial spin-orbit integrals, see Ref. [123]. Both the one- and the two-electron integrals are calculated, the latter is stored on the file `AO2SOINT`.

One-electron Integral: $\sum_A Z_A \left\langle \chi_\mu \left| \frac{\mathbf{l}_A}{r_A^3} \right| \chi_\nu \right\rangle$ where Z_A is the charge of nucleus A and the summation runs over all nuclei of the molecule.

Property labels: `X1SPNORB`, `Y1SPNORB`, `Z1SPNORB`

Two-electron Integral: $\left\langle \chi_\mu \chi_\nu \left| \frac{1_{12}}{r_{12}^3} \right| \chi_\rho \chi_\sigma \right\rangle$

Property labels: `X2SPNORB, Y2SPNORB, Z2SPNORB`

.SQHDOR Square, non-symmetrized half-differentiated overlap integrals with respect to geometric distortions, see Ref. [118]. Differentiation on the ket-vector.

Integral: $\left\langle \chi_\mu \mid \frac{\partial \chi_\nu}{\partial R_{ab}} \right\rangle$

Property label: `SQHDRabc`, where `abc` is the number of the symmetry-adapted coordinate being differentiated.

.SUPONL Only calculate the supermatrix. Requires the presence of the two-electron integral file.

.SUSCGO Diamagnetic magnetizability integrals calculated without the use of London atomic orbitals. The choice of gauge origin can be controlled by the keyword **.GAUGEO**.

Integral: $\frac{1}{4} \left\langle \chi_\mu \left| r_O^2 I - \mathbf{r}_O \mathbf{r}_O^T \right| \chi_\nu \right\rangle$

Property labels: `XXSUSCGO, XYSUSCGO, XZSUSCGO, YYSUSCGO, YZSUSCGO, ZZSUSCGO`

.THETA Traceless quadrupole moment integrals as defined by Buckingham [43].

Integral: $\frac{1}{2} \left\langle \chi_\mu \left| 3\mathbf{r}\mathbf{r}^T - r^2 I_3 \right| \chi_\nu \right\rangle$

Property labels: `XXTHETA , XYTHETA , XZTHETA , YYTHETA , YZTHETA , ZZTHETA`

.THIRDM Third-moment integrals.

Integral: $\left\langle \chi_\mu \left| \mathbf{r}^3 \right| \chi_\nu \right\rangle$

Property labels: `XXX 3MOM, XXY 3MOM, XXZ 3MOM, XYY 3MOM, XYZ 3MOM, XZZ 3MOM,` `YYY 3MOM, YYZ 3MOM, YYZ 3MOM, ZZZ 3MOM.`

.U12INT Calculation of two-electron integrals over $[T1, r_{12}]$.

.U21INT Calculation of two-electron integrals over $[T2, r_{12}]$.

.WEINBG
    `READ (LUCMD,*) BGWEIN`
    Read in the square of the sin of the Weinberg angle appearing in the definition of parity-violating integrals, see **.PVIOLA**. The Weinberg angle factor will if this keyword is used be set to `[1-4*BGWEIN]`.

.XDDXR3 Direct perturbation theory paramagnetic spin–orbit like integrals.

Integral:

Property labels: `ALF abcd`, where a is ????.

22.2.3 One-electron integrals: `*ONEINT`

Directives affecting the one-electron undifferentiated Hamiltonian integral calculation appear in the `*ONEINT` section.

`.CAVORG`

> `READ (LUCMD,*) (CAVORG(I), I = 1, 3`
>
> Read one more line containing the origin to be used for the origin of the cavity in self-consistent reaction field calculations. The default is that this origin is chosen to be the center of mass of the molecule.

`.NOT ALLRLM` Save only the totally symmetric multipole integrals calculated in the solvent run on disc. Default is that multipole integrals of all symmetries are written disc. May be used in calculations of the energy alone in order to save disc space.

`.PRINT`

> `READ (LUCMD,*) IPRONE`
>
> Set print level during the calculation of one-electron Hamiltonian integrals. Read one more line containing print level. Default is the value of `IPRDEF` from the `**INTEGRALS` input module.

`.SKIP` Skip the the calculation of one-electron Hamiltonian integrals. Mainly for debugging purposes.

`.SOLVEN`
> `READ (LUCMD,*) LMAX`

> > Integral: $\left\langle \chi_\mu \left| x^i y^j z^k \right| \chi_\nu \right\rangle$ for all integrals where $i + j + k \leq$ `LMAX`.
> >
> > Property label: Not extractable. Integrals written to file `AOSOLINT`

> Calculate the necessary integrals needed to model the effects of a dielectric medium by a reaction-field method as described in Ref. [94]. Read one more line containing maximum angular quantum number for the multipole integrals used for the reaction field.

22.2.4   General: *READIN

Directives to control the reading of the molecule input appear in the *READIN section.

.CM FUN

>   READ (LUCMD,*) LCMMAX, CMSTR, CMEND

> Use Rydberg basis functions (center of mass functions) as suggested by Kaufman *et al.* [124]. LCMMAX denoted the maximum quantum number of the Rydberg functions, basis functions for all quantum up to and including LCMMAX will be generated (s=0, p=1 etc.) CMSTR and CMEND are the half-integer start- and ending quantum number for the Rydberg basis functions. The basis functions will be placed at the position of a dummy center indicated as X in the MOLECULE.INP file. The charge of the ion-core is determined by the keyword .ZCMVAL. If no center named X is present in the MOLECULE.INP file, this input will be ignored.

.MAXPRI

>   READ (LUCMD,*) MAXPRI

> Set maximum number of primitives in any contraction. Read one more line containing number. Default is 25, except for the Cray-T3D/E, where the default is 14.

.OLDNORM Use the normalization scheme for spherical AO's used in DALTON 1.0. From DALTON 1.1, all components of spherical AO's are normalized to 1. This was not the case in DALTON 1.0, and this option is needed to read MO's correctly from an old file generated by DALTON 1.0 if spherical AO's were used. (Cartesian AO's are still not all normalized to 1 for d-orbitals and higher l's.)

.PRINT

>   READ (LUCMD,*) IPREAD

> Set print level for input processing. Read one more line containing print level. Default is the IPRDEF from the **INTEGRALS input module.

.SYMTHR

>   READ (LUCMD,*) TOLLRN

> Read threshold for considering atoms to be related by symmetry. Used in the automatic symmetry detection routines. Default is $5.0 \cdot 10^{-6}$.

.UNCONT Force the program to use the basis set as a primitive (completely decontracted) set.

.WRTLIN Write out the lines read in during the input processing of the MOLECULE.INP file. Primarily for debugging purposes or for analysing input errors in the MOLECULE.INP file.

```
.ZCMVAL
     READ (LUCMD,*) ZCMVAL
```

Read the charge of the center of the Rydberg basis functions specified by the `.CM FUN` keyword. Default is a charge of one.

22.2.5 Integral sorting: *SORINT

Affects the sorting of the two-electron integrals. Note that in order for SIRIUS to use the sorted integrals generated with the keyword `.SORT I`, the keyword `.PRESORT` has to used in the `*TRANSFORMATION` input module of the `**WAVE FUNCTIONS` input section. For standard calculation you will in general not need to change any of the default settings.

`.DELAO` Delete AOTWOINT file from Hermit after the integrals have been sorted.

```
.INTSYM
     READ (LUCMD, *) ISNTSYM
```

Symmetry of the two-electron integrals that are to be sorted. Default is totally-symmetric two-electron integrals (`ISNTSYM` = 1).

```
.IO PRI
     READ (LUCMD, *) ISPRFIO
```

Set the print level in the fast-I/O routines. Default is a print level of 0.

```
.KEEP
     READ (LUCMD, *) (ISKEEP(I),I=1,8)
```

Allowed values: 0 and 1. A value of ”1” indicates that the basis functions of this symmetry will not be used and the integrals with these basis functions are omitted.

```
.PRINT
     READ (LUCMD, *) ISPRINT
```

Print level in the integral sorting routines.

```
.THRQ
     READ (LUCMD, *) THRQ2
```

Threshold for setting an integral to zero. By default this threshold is 1.0D-15.

22.2.6 Construction of the supermatrix file: *SUPINT

Directives affecting the construction of the supermatrix file is given in the `*SUPINT` section.

.NOSYMM No advantage is taken of integral symmetry in the construction of the supermatrix file. This may increase the disc space requirements as well as CPU time. Since ABACUS do not use the supermatrix file (which it in the current version does not), this keyword is mainly for debugging purposes.

.PRINT

        READ (LUCMD,*) IPRSUP

Set the print level during the construction of the supermatrix file. Read one more line containing the print level. Default is the value of `IPRDEF` in the `**INTEGRALS` input module.

.SKIP Skip the construction of the supermatrix file. An alternative keyword is `.NOSUP` in the `**INTEGRALS` input module.

.THRESH

        READ (LUCMD,*) THRSUP

Threshold for the supermatrix integrals. Read one more line containing the threshold. Default is the same as the threshold for discarding the two-electron integrals (see the chapter describing the MOLECULE input format, Ch. 23).

22.2.7 Two-electron integrals using TWOINT: *TWOINT

Directives affecting the two-electron undifferentiated Hamiltonian integral calculation appear in the *TWOINT section.

.ICEDIF

        READ (LUCMD,*) ICDIFF,IEDIFF

Screening threshold for Coulomb and exchange contributions to the Fock matrix in direct and parallel calculations. The thresholds for the integrals are ten to the negative power of these numbers. By default the same screening threshold will be used for Coulomb and exchange contribution which will change dynamically as the wave function converges more and more tightly.

.IFTHRS

        READ (LUCMD,*) IFTHRS

Screening threshold used in direct and parallel calculations. The integral threshold will be ten to the negative power of this number. The default is that this value will change dynamically as the wave function converges more and more tightly.

.PANAS Calculates scaled two-electron integrals as proposed by Panas as a simple way of introducing electron correlation in calculations of molecular energies [125].

.PRINT

    READ (LUCMD, *) IPRINT, IPRNTA, IPRNTB, IPRNTC, IPRNTD

    Set print level for the derivative integral calculation of a particular shell quadruplet. Read one more line containing print level and the four shell indices. The print level is changed from the default for this quadruplet only.

.RETURN Stop after the shell quadruplet specified under **.PRINT** above. Mainly for debugging purposes.

.SKIP Skip the calculation of two-electron Hamiltonian integrals. An alternative keyword is **.NOTWO** in the ****INTEGRALS** input module.

.SOFOCK Construct the Fock matrix in symmetry-orbital basis during a direct or parallel calculation. Currently not active.

.THRFAC Not used in DALTON.

.TIME Provide detailed timing breakdown for the two-electron integral calculation.

22.2.8 Two-electron integrals using ERI: *ER2INT

Directives controlling the two-electron integral calculation with ERI are specified in the *ER2INT section. By default ERI is only used for integral-direct coupled cluster calculations. However, ERI can also be invoked by specifying the **.RUNERI** keyword in the ****DALTON INPUT** input section. Note that DALTON will automatically use TWOINT for all integrals not available in ERI.

.AOBTCH

    READ (LUCMD,*) IAOBCH

    Only integrals with the first integral index belonging to AO batch number **IAOBCH** will be calculated.

.BUFFER

    READ (LUCMD,*) LBFINP

    This option may be used to set the buffer length for integrals written to disk. For compatibility with TWOINT, the default buffer length is 600. Longer buffer lengths may give more efficient I/O.

.DISTRI Use distributions for electron 1. Must be used in connection with the keywork **.SELCT1**.

.DISTST Test the calculation of two-electron integrals using distributions.

.DOERIP Use the ERI integral program for the calculation of two-electron integrals instead of TWOINT.

.EXTPRI

    `READ (LUCMD,*) IPROD1, IPROD2`

Full print for overlap distribution (OD) classes `IPROD1` and `IPROD2`.

.GENCON Treat all AOs as generally contracted during integral evaluation.

.GRDZER During evaluation of the molecular gradient, the gradient is set to zero upon each entry into **ERIAVE** (for debugging).

.INTPRI Force the printing of calculated two-electron integrals.

.INTSKI Skip the calculation of two-electron integrals in the ERI calculation.

.MAXDIS

    `READ (LUCMD,*) MAXDST`

Read in the maximimum number of integral distributions calculated in each call to the integral code. Default value is 40.

.MXBCH

    `READ (LUCMD,*) MXBCH`

Read in the maximum number of integral batches to be treated simultaneously, and thus determines the vector lengths. Default value is 1000000000.

.NCLERI Calculate only integrals with non-classical contributions.

.NEWCR1 Use an old transformation routine for the generation of the Cartesian integrals for electron 1.

.NOLOCS Do not use local symmetry during evaluation.

.NONCAN Do not sort integral indices in canonical order. This option is applicable only for undifferentiated integrals written fully to disk (without the use of distributions). This option may save some time but will lead to incorrect results whenever canonical ordering is assumed.

.NO12GS During sorting of OD batches, treat batches containing one and two distinct AOs as equivalent.

.NOPS12 Do not assume permutational symmetry between the two electrons.

.NOPSAB Do not assume permutational symmetry between orbitals of electron 1.

.NOPSCD Do not assume permutational symmetry between orbitals of electron 2.

.NOSCRE Do not do integral screening before integrals are written to disk.

.NOWRIT Do not write integrals to disk.

.NSPMAX

```
READ (LUCMD,*) NSPMAX
```

Allows for the reduction of the number of symmetry generations for each basis function in order to reduce memory requirements. Mainly for debugging purposes, do not use.

.OFFCNT During sorting of OD batches, treat batches containing one and two distinct AO centers as equivalent.

.PRINT

```
READ (LUCMD,*) IPRERI, IPRNT1, IPRNT2
```

Print level for ERI. By giving numbers different from zero for IPRNT1 and IPRNT2, extra print information may be given for these overlap distributions, after which the program will exit.

.RETURN Stop after the shell quadruplet specified under .PRINT above. Mainly for debugging purposes.

.SELCT1

```
READ (LUCMD,*) NSELCT(1)
READ (LUCMD,*) (NACTAO(I,1),I=1,NSELCT(1))
```

Calculate only integrals containing indices NACTAO(I,1) for the first AO.

.SELCT2 Same as .SELCT1 but for the second AO.

.SELCT3 Same as .SELCT1 but for the third AO.

.SELCT4 Same as .SELCT1 but for the fourth AO.

.SKIP Skip the calculation of two-electron Hamiltonian integrals. An alternative keyword is .NOTWO in the **INTEGRALS input module.

.TIME Provide detailed timings for the two-electron integral calculation in ERI.

.WRITEA Write all integrals to disk without any screening.

Chapter 23

MOLECULE **input style**

The MOLECULE-style input is originally based on the input for the MOLECULE integral program by Almlöf [126]. However, there are few remnants of the original input structure. For users of earlier releases of the Dalton program, note should be taken of the fact that the input structure for the MOLECULE file has undergone major changes, however, backward compatibility has in most cases been retained[1]. The program supports both Cartesian and Z-matrix input of the molecular coordinates. However, the Z-matrix input provided is only a convenient way of describing the molecular geometry, as the Z-matrix is converted to Cartesian coordinates, which are then used in the subsequent calculations. Note that the Z-matrix input can only be used together with the basis set library ("BASIS" in first line).

The program includes an extensive basis set library, which are described below. There are additional possibilities for choosing the number of primitive and contracted orbitals to be used with the Atomic Natural Orbital (ANO) basis sets and the "Not Quite van Duijneveldt" (NQvD) basis sets. The large number of different basis sets provided is in part related to the variety of molecular properties with very different basis set requirements that can be calculated with DALTON. Most of the basis sets have been downloaded from the EMSL basis set library service

(http://www.emsl.pnl.gov:2080/forms/basisform.html) [127]. Only six of the basis sets were obtained from different sources:

- The ano-1, ano-2, ano-3, ano-4 and Sadlej-pVTZ basis sets which where downloaded from the MOLCAS home-page (http://garm.teokem.lu.se/MOLCAS/).

- The NQvD (The Not Quite van Duijneveldt) basis sets were constructed by Knut Fægri for this distribution [128].

---

[1]The two exceptions to the backward are that when using the **ATOMBASIS** keyword, the basis set name has to be preceded by "Basis=". When specifying the Cartesian coordinates of the atoms, it is no longer required that the coordinates start at position 5. However, blanks are no longer allowed in the names of atoms, and the atom names are still restricted to 4 characters.

We note that each file containing a given basis set in the BASIS directory contains the proper reference to be used when doing a calculation with a given basis set. For convenience we also list these references with the basis sets in Section 23.6.

The description of the MOLECULE input is divided into five parts. Section 23.1 describes the general section of the molecule input, section 23.2 describes the Cartesian coordinate input, section 23.3 describes the Z-matrix input, and finally Section 23.4 describes the basis set library. Section 23.6 lists the basis sets (including references and supported elements) in the basis set library.

23.1   General MOLECULE input

In the general input section of the MOLECULE input file, we will consider such information as molecular symmetry, number of symmetry distinct atoms, generators of a given molecular point group, and so on. This information usually constitutes the four/five first lines of the input.

The input is best described by an example. The following is the first lines of an input for tetrahedrane, treated in C_{2v} symmetry, with a 4-31G** basis. The line numbers are for convenience in the subsequent input description and should *not* appear in the actual input. Note also that in order to fit the example across the page some liberties have been taken with column spacings.

```
1:INTGRL
2:          Tetrahedrane, Td_symmetric geometry
3:                 4-31G** basis
4:Atomtypes=2 Generators=2 X Y Integrals=1.00D-15
```

We now define the input line-by-line. The `FORMAT` is given in parenthesis.

1 The word `INTGRL` (`A6`).

2-3 Two arbitrary title lines (`A72`).

4 General instructions about the molecule.

This line is keyword-driven. The general structure of the input is `Keyword=`. The input is case sensitive, but DALTON will recognize the keywords whether specified with only three characters (minimum) or the full name (or any intermediate option). The order of the keywords is arbitrary. The following keywords are recognized for this line:

`Angstrom` Indicates that the atomic Cartesian coordinates are given in Ångström, and not in bohr (atomic units) which is the default.

Atomtypes (`Integer`). *This keyword is required.* Number of atom types (number of atoms specified in separate blocks). For a Z-matrix input this will be the total number of atoms in the molecule, the Z-matrix module will then extract the number of atom types.

Cartesian Indicates that a Cartesian Gaussian basis set will be used in the calculations.

Charge (`Integer`). The charge of the molecule. Will be used by the program to determine the Hartree–Fock occupation.

Generators (`Integer+Character`). Number of symmetry generators. If this keyword is not specified (and `Nosymmetry` not invoked) the automatic symmetry detection routines of the program will be invoked. Symmetry can be turned off (needed for instance if starting a walk at a highly symmetric structure which one knows will break symmetry) using the keyword `Nosymmetry`. DALTON is restricted to the Abelian subgroups of D_{2h}, and thus there can be 1 to 3 generating elements.

The number of generators is followed the equally many blocks of characters specifying which Cartesian axis change sign during each of the generators. `X` is reflection in the yz-plane, `XY` is rotation about the z-axis, and `XYZ` denotes inversion. Due to the handling of symmetry in the program, it is recommended to use mirror planes as symmetry generating elements if possible.

Integrals (`Real`). Indicates the threshold for which integrals smaller than this will be considered to be zero. If not specified, a threshold of 1.0D-15 will be used. A threshold of 1.0D-15 will give integrals correct to approximately 1.0D-13.

Nosymmetry Indicates that the calculation is to be run without the use of point-group symmetry. Automatic symmetry detection will also be disabled.

Own Indicates that a user-supplied scheme for generating transformed angular momentum basis functions will be used.

Spherical Default. Indicates that a spherical Gaussian basis set will be used in the calculations.

Note that if one wants to use the basis set library, there are two options. One option is to use a common basis set for the entire molecule in which the first line should be replaced by two lines, which for a calculation using the 4-31G** basis would look like:

```
1:BASIS
2:4-31G**
```

This option will not be active with customizable basis sets like the ANO or NQvD sets.

Alternatively you may specify different basis sets for different atoms, in which case the first line should read

```
1:ATOMBASIS
```

The fourth line (fifth in a calculation using the basis set library) looks a bit devastating. However, for ordinary Hartree–Fock or MP2 calculations, only the number of different atom types and the charge need to be given (if the molecule is charged), as symmetry and Hartree–Fock occupation will be taken care of by the program. Thus this line could in the above example be reduced to

```
4:Atomtypes=2
```

or even more concisely (though not more readable) as

```
4:Ato=2
```

Let us finally give some remarks about the symmetry detection routines. These routines will detect any symmetry of a molecule by explicit testing for the occurrence of rotation axes, mirror planes and center of inversion. The occurrence of a symmetry element is tested in the program against a threshold which may be adjusted by the keyword .SYMTHR in the *READIN input section. By default, the program will require geometries that are correct to the sixth decimal place in order to detect all symmetry elements.

The program will translate and rotate the molecule into a suitable reference geometry before testing for the occurrence of symmetry operations. The program will not, due to the handling of symmetry in the program, transform the molecule back to original input coordinates. Furthermore, if there are symmetry equivalent nuclei, these will be removed from the input, and a new, standardized molecule input file will be generated and used in subsequent iterations of for instance a geometry optimization. This standardized input file (including basis set) is printed to the file DALTON.BAS, which is among the files copied back after the end of a calculation.

DALTON can only take advantage of point groups that are subgroups of D_{2h}. If symmetry higher than that is detected, the program will use the highest common subgroup of the symmetry group detected and D_{2h}.

We recommend that the automatic symmetry detection feature is not used when doing MCSCF calculations, as symmetry generators and their order in the input determines the order of the irreducible representations needed when specifying active spaces. Thus, for MCSCF calculations we recommend that the symmetry is explicitly specified through the appropriate symmetry generators, as well as the explicit Hartree–Fock occupation numbers.

23.2   Cartesian geometry input

Assuming that we have given the general input as indicated above, we now want to specify the spatial arrangements of the atoms in a Cartesian coordinate system. We will also for sake of illustration assume that we have given explicitly the generators of the point group to be used in the calculation (in this case C_{2v}, with the yz- and xz-planes as mirror planes).

In tetrahedrane we will have two different kinds of atoms, carbon and hydrogen, as indicated by the number 2 on the fourth (fifth) line of the input. We will also assume that we punch the basis set ourselves, in order to present the input format for the basis set.

For tetrahedrane, the input would then look like

```
 5:Charge=6.0 Atoms=2 Blocks=3 1 1 1
 6:C1    1.379495419            .0                  0.975450565
 7:C2     .0                    1.379495419         -.975450565
 8:    8    3
 9:486.9669    .01772582
10:73.37109    .1234779
11:16.41346    .4338754
12:4.344984    .5615042
13:8.673525            -.1213837
14:2.096619            -.2273385
15:.6046513            1.185174
16:.1835578                    1.00000
17:    4    2
18:8.673525    .06354538
19:2.096619    .2982678
20:.6046513    .7621032
21:.1835578            1.000000
22:    1    1
23:0.8         1.0
24:Charge=1.0 Atoms=2 Blocks=2 1 1
25:H1    3.020386510           .0                  2.1357357837
26:H2     .0                   3.020386510         -2.1357357837
27:    4    2
28:18.73113    .03349460
29:2.825394    .2347270
30:.6401218    .8137573
31:.1612778            1.000000
32:    1    1
```

```
33:0.75      1.0
```

The different lines are:

5 This line is keyword-driven. The general structure of the input is `Keyword=`. The input is case sensitive, but it will recognize the keywords whether specified with only three characters (minimum) or the full name (or any intermediate option). The order of the keywords is arbitrary. The following keywords are recognized for this line:

Atoms (Integer). Number of *symmetry-distinct* atoms of this type (or, if the symmetry detection routines are being used, all atoms of this kind).

Basis (Character). If `ATOMBASIS` has been specified, the keyword is required, and have to be followed by the name of the basis set that is to be used for this group of atoms, *e.g.* `Basis=6-31G**`. By specifying `Basis=pointcharge`, the atoms in this block will be treated as point charges, that is, having only a charge but no basis functions attached to them. Effective core potentials (ECP) can also by used by specifying `Basis=ECP`.

Blocks (Integers). Maximum angular quantum number + 1 used in the basis set for this atom type ($s = 1$, $p = 2$, etc.). Ignored if library basis sets are being used (`BASIS` or `ATOMBASIS` in first line). This number is followed by one integer for each angular momentum used in the basis, indicating the number of groups (blocks) of generally contracted functions of angular quantum number `I-1`. Ignored if the basis set library is used.

It is noteworthy that DALTON collects all basis functions into one such shell, and evaluates all integrals arising from that shell simultaneously, and the memory requirements grow rapidly with the number of basis functions in a shell (note for instance that four g functions actually are 36 basis functions, as there are 9 components of each g function). Memory requirements can therefore be reduced by splitting basis functions of the quantum number into different blocks. However, this will decrease the performance of the integral calculation.

Charge (Real). *This keyword is required.* Charge of this atom or point charge.

Pol (Integer+real). This keyword adds single, primitive basis function of a given quantum number (quantum number + 1 given in the input) and a given exponent. An arbitrary number of polarization functions can be given. For instance, we can add a p function with exponent 0.05 and a d function with exponents 0.6 we can write `Pol 2 0.05 3 0.6`.

Set (Integer). Indicates whether the basis set specified is the ordinary orbital basis or the auxiliray basis set needed for instance in certain r12 calculations, see

Sec. 23.5. The keyword is only active when the keyword `.AUXBAS` has been specified in the `**INTEGRALS` input section.

6 NAME X Y Z Isotope=18

> **NAME** Atom name. A different name should be used for each atom of the same type, although this is not required. Note that only the first four characters of the atom name will be used by the program, and that no spaces are allowed in naming the atom.
>
> **X** x-coordinate (in atomic units, unless Ångström has been requested on line 4 of the input).
>
> **Y** y-coordinate.
>
> **Z** z-coordinate.
>
> **Isotope=** Specify the atomic mass of the nucleus (closest integer number). By default the mass of the most abundant isotope of the element will be used. When automatic symmetry detection is used, the program will distinguish between different nuclei if they have different atomic mass number. A calculation of HDO would thus be run in C_s symmetry.

The Cartesian coordinates may be given in free format.

7 This is the other symmetry-distinct center of this type.

8 FRMT, NPRIM, NCONT, NOINT (A1,I4,2I5).

> **FRMT** A single character describing the input format of the basis set in this block. The default format is (`8F10.4`) which will be used if `FRMT` is left blank. In this format the first column is the orbital exponent and the seven last columns are contraction coefficients. If no numbers are given, a zero is assumed. If more than 7 contracted functions occur in a given block, the contraction coefficients may be continued on the next line, but the first column (where the orbital exponents are given) must then be left blank.
>
> An `F` or `f` in the first position will indicate that the input is in free format. This will of course require that all contraction coefficients need to be typed in, as all numbers need to be present on each line. However, note that this options is particularly handy together with completely decontracted basis sets, as described below. Note that the program reads the free format input from an internal file that is 80 characters long, and no line can therefore exceed 80 characters.
>
> One may also give the format `H` or `h`. This corresponds to high precision format (`4F20.8`), where the first column again is reserved for the orbital exponents,

and the three next columns are designated to the contraction coefficients. If no number is given, a zero is assumed. If there are more than three contracted orbitals in a given block, the contraction coefficients may be continued on the next line, though keeping the column of the orbital exponents blank.

NPRIM Number of primitive Gaussians in this block.

NCONT Number of contracted Gaussians in this block. If a zero is given, an uncontracted basis set will be assumed, and only orbital exponents need to be given.

9 EXP, (CONT(I), I=1,NCONT)

EXP Exponent of this primitive.

CONT(I) Coefficient of this primitive in contracted function I.

We note that the format of the orbital exponents and the contraction coefficients are determined from the value of FRMT defined on line 8.

10-16 These lines complete the specification of this contraction block: the s basis here.

17-21 New contraction block (see lines 8 and 9 above).

22-23 New contraction block.

24-33 Specifies a new atom type: coordinates and basis set.

23.3   Z-matrix input

The Z-matrix input provided with DALTON is quite rudimentary, and common options like parameter representations of bond length and angles as well as dummy atoms are not provided. Furthermore, the Z-matrix input is not used in the program, but instead immediately converted to Cartesian coordinates which are then used in the subsequent calculation. Another restriction is that if Z-matrix input is used, one cannot punch ones own basis set, but must instead resort to one of the basis sets provided with the basis set library (i.e. BASIS in first line). Finally, you cannot explicitly specify higher symmetry in input line 4 when using Z-matrix input, you can only request no symmetry (C_1) with Nosymmetry or allow DALTON to detect symmetry automatically.

The input format is free, with the restriction that the name of each atom must be given a space of 4 characters, and none of the other input variables needed must be placed in these positions.

The program will use Z-matrix input if there is the word "ZMAT" in the first four position of line 6 in the molecule input. The following NONTYP lines contain the Z-matrix specification for the NONTYP atoms.

A typical Z-matrix input could be:

```
BASIS
6-31G**
   Test Z-matrix input of ammonia
   6-31G** basis set
Atomtypes=4
ZMAT
N   1 7.0
H1  2 1 1.0116 1.0
H2  3 1 1.0116 2 106.7 1.0
H3  4 1 1.0116 2 106.7 3 106.7 1 1.0
```

The five first lines should be familiar by now, and will be discussed no further here. The special 6'th line tells that this is Z-matrix input. The Z-matrix input starts on line 7, and on this first Z-matrix line only the atom name, a running number and the charge of the atom is given. The running number is only for ease of reference to a given atom, and is actually not used within the program, where any reference to an atom, is the number of the atom consecutively in the input list.

The second Z-matrix line consists of the atom name, a running number, the number of the atom to which this atom is bonded with a given bond length in Ångströms, and then finally the charge of this atom.

The third Z-matrix line is identical to the second, except that an extra atom number, to which the two first atoms on this line is bonded to with a given bond angle in degrees.

On the fourth Z-matrix line yet another atom has been added, and the position of this atom relative to the three previous ones on this line is dependent upon on an extra number inserted just before the nuclear charge of this atom. If the next to last number is a 0, the position of this atom is given by the dihedral angle (`A1,A2,A3,A4`) in degrees, where `Ai` denotes atom `i`. If, on the other hand, this next to last number is ± 1, the position of the fourth atom is given with respect to two angles, namely (`A1,A2,A3`) and (`A2,A3,A4`). The sign is to be +1 if the triple product $\overrightarrow{(A_2A_1)} \cdot \left[\overrightarrow{(A_2A_3)} \times \overrightarrow{(A_2A_4)} \right]$ is positive.

23.4   Using basis set libraries

The use of predefined basis sets is indicated by the word `BASIS` or `ATOMBASIS` on the first line of the molecular input. (If you want Z-matrix input you must use `BASIS`.)

The specified basis set(s) are searched for in the following directories:

- all user specified basis set directories (with `dalton -b dir1 -b dir2 ...`

- the job directory

- the basis set library supplied with DALTON.

If `BASIS` is used, a common basis set is used for all atoms in the molecule, and the name of this basis set is given on the second line. If we want to use one basis set for all the atoms in a molecule, the molecule input file can be significantly simplified, as we may delete all the input information regarding the basis set. Thus, the input in the previous section for tetrahedrane with the 6-31G** basis will, if the basis set library is used, be reduced to:

```
 1:BASIS
 2:6-31G**
 3:          Tetrahedrane, Td_symmetric geometry
 4:                  4-31G** basis
 5:Atomtypes=2 Generators=2 X Y Integrals=1.00D-15
 6:Charge=6.0 Atoms=2
 7:C1    1.379495419           .0                0.975450565
 8:C2     .0               1.379495419          -.975450565
 9:Charge=1.0 Atoms=2
10:H1    3.020386510           .0                2.1357357837
11:H2     .0               3.020386510          -2.1357357837
```

The use of the basis set library is indicated by the presence of the `BASIS` word in the beginning of MOLECULE-file instead of `INTGRL`.

An alternative approach would be to use different basis sets for different atoms, *e.g.* the concept of locally dense basis sets introduced in NMR calculations by Chesnut *et al.* [129]. This is for instance also required when using the ANO or NQvD basis sets. Another option is to use standards basis sets from the basis set library and add your own sets of diffuse, tight or polarizing basis functions. Returning to tetrahedrane, we could for instance use the 6-31G* basis set for carbon and the 4-31G** basis set for hydrogen. This could be achieved as

```
 1:ATOMBASIS
 2:          Tetrahedrane, Td_symmetric geometry
 3:    Mixed basis (6-31G* on C and 4-31G** on H)
 4:Atomtypes=2 Generators=2 X Y Integrals=1.00D-15
 5:Charge=6.0 Atoms=2 Basis=6-31G*
 6:C1    1.379495419           .0                0.975450565
 7:C2     .0               1.379495419          -.975450565
 8:Charge=1.0 Atoms=2 Basis=4-31G Pol 2 0.75D0
 9:H1    3.020386510           .0                2.1357357837
10:H2     .0               3.020386510          -2.1357357837
```

Thus, when using `ATOMBASIS` the name of the basis set for a given set of identical atoms is given on the same line as the nuclear charge, indicated by the keyword "Basis=".

The string `Pol` denotes that the rest of the line specifies diffuse, tight or polarizing functions, all which will be added as segmented basis functions. For each basis function, its "angular momentum" $(l+1)$ and its exponent must be given. Thus, in the above input we indicate that we add a p function with exponent 0.75 to the hydrogen basis set. The order of these functions are arbitrary (that is, a p function can be given before an s function and so on).

Augmenting the correlation-consistent sets of Dunning [68] is a straight-forward process in DALTON. The aug-cc-pVXZ and aug-cc-pCVXZ basis sets are extended in an even-tempered manner (in the manner of Dunning [68]) by including a 'd-', 't-' or 'q-' prefix to give doubly, triply or quadruply augmented basis sets, respectively. For example, specifying t-aug-cc-pVDZ will produce a triply augmented cc-pVDZ basis. The aug-ecp basis sets may be augmented in the same manner by using the same prefixes, eg d-aug-ecp. Note that these basis sets are not listed explicitly in the basis library directory, but are automatically generated within DALTONfrom the respective aug-cc-pVXZ and aug-ecp basis sets.

The ANO basis sets require that you give the number of contracted functions you would like to use for each of the primitive sets defined in the basis sets. Thus, assuming we would like to simulate the 6-31G** basis set input using an ANO basis set but with the polarization functions of the 6-31G** set, this could be achieved through an input like

```
 1:ATOMBASIS
 2:        Tetrahedrane, Td_symmetric geometry
 3:    Mixed basis (6-31G* on C and 4-31G** on H)
 4:Atomtypes=2 Generators=2 X Y Integrals=1.00D-15
 5:Charge=6.0 Atoms=2 Basis=ano-1 3 2 0 0 Pol 3 0.8
 6:C1    1.379495419          .0                0.975450565
 7:C2     .0               1.379495419         -.975450565
 8:Charge=1.0 Atoms=2 Basis=ano-1 2 0 0 Pol 2 0.75D0
 9:H1    3.020386510          .0                2.1357357837
10:H2     .0               3.020386510         -2.1357357837
```

This input will give a [3s2p0d0f] ANO basis set on carbon, with a polarizing d function with exponent 0.8, and a [2s0p0d] ANO basis set on hydrogen with a polarizing p function with exponent 0.75 as above.

Note that the number of contracted functions in the ANO set has to be given for all primitive blocks, even though you do not want any contracted functions of a given quantum number. Here also, `Pol` separates the number of contracted functions from polarization functions.

The NQvD basis set [128] was constructed in order to provide, in electronic form, a basis set compilation very similar to original set of van Duijneveldt [130]. The sets are in general as good, or slightly better, than the original van Duijneveldt basis, with only minor changes in the orbital exponents.

In the NQvD basis set, you need not only to pick the number of contracted functions, but also your primitive set. The contracted basis set will be constructed contracting the (NPRIM-NCONT + 1) tightest functions with contraction coefficients based on the eigenvectors from the atomic optimization, keeping the outermost orbitals uncontracted.

NOTE: As is customary, the orbital exponents of all hydrogen basis functions are automatically multiplied by a factor of 1.44.

Thus, an input for tetrahedrane employing the NQvD basis set might look like

```
 1:ATOMBASIS
 2:        Tetrahedrane, Td_symmetric geometry
 3:   Mixed basis (6-31G* on C and 4-31G** on H)
 4:Atomtypes=2 Generators=2 X Y Integrals=1.00D-15
 5:Charge=6.0 Atoms=2 Basis=NQvD 8 4 3 2 Pol 3 0.8D0
 6:C1    1.379495419          .0               0.975450565
 7:C2     .0                1.379495419        -.975450565
 8:Charge=1.0 Atoms=2 Basis=NQvD 4 2 Pol 2 0.75D0
 9:H1    3.020386510          .0               2.1357357837
10:H2     .0                3.020386510        -2.1357357837
```

This input will use an (8s4p/4s) primitive basis set on carbon and hydrogen respectively, contracting it to a [3s2p/2s] set. The polarization functions should not require further explanation at this stage.

The only limitations to the use of polarization functions when ATOMBASIS is used, is that the length of the line must note exceed 80 characters. If that happens, we recommend collecting a standard basis set from the file DALTON.BAS, and then adding functions to this set.

23.5   Auxiliary basis sets

It is possible to specify more than one basis set. For example, by typing

```
 1:BASIS
 2:4-31G** 6-311++G(3df,3pd)
```

the basis set 6-311++G(3df,3pd) will be used as an auxiliary basis. When using an auxiliary basis, each atom line must contain a basis-set identifier, which at present may take the values

`Set=1` (orbital basis) or `Set=2` (auxiliary basis). Basis sets with `Set=1` must be read first, basis sets with `Set=2` thereafter. The above also applies to the `ATOMBASIS` and `INTGRL` input modes. Examples are provided by the following input files:

```
 1:BASIS
 2:cc-pVDZ cc-pCVQZ
 3:Direct MP2-R12/cc-pVDZ calculation on H2O
 4:Auxiliary basis: cc-pCVQZ
 5:Atomtypes=1 Generators=1 X
 6:Charge=8.0 Atoms=1 Set=1
 7:O         .000000000000000      .000000000000000   -0.124309000000000        *
 8:Charge=1.0 Atoms=1 Set=1
 9:H      1.427450200000000      .000000000000000    0.986437000000000        *
10:Charge=8.0 Atoms=1 Set=2
11:O         .000000000000000      .000000000000000   -0.124309000000000        *
12:Charge=1.0 Atoms=1 Set=2
13:H      1.427450200000000      .000000000000000    0.986437000000000        *
```

```
 1:ATOMBASIS
 2:Direct MP2-R12/cc-pVDZ calculation on H2O
 3:Auxiliary basis: cc-pCVQZ
 4:Atomtypes=4 Generators=1 X
 5:Charge=8.0 Atoms=1 Set=1 Basis=cc-pVDZ
 6:O         .000000000000000      .000000000000000   -0.124309000000000        *
 7:Charge=1.0 Atoms=1 Set=1 Basis=cc-pVDZ
 8:H      1.427450200000000      .000000000000000    0.986437000000000        *
 9:Charge=8.0 Atoms=1 Set=2 Basis=cc-pCVQZ
10:O         .000000000000000      .000000000000000   -0.124309000000000        *
11:Charge=1.0 Atoms=1 Set=2 Basis=cc-pCVQZ
12:H      1.427450200000000      .000000000000000    0.986437000000000        *
```

Note that the keyword `.AUXBAS` must be specified in the `**INTEGRALS` input section to be able to read the above inputs.

23.6 The basis sets supplied with DALTON

As mentioned above, all the basis sets supplied with this distribution of the DALTON program — with the exception of the ANO, Sadlej-pVTZ and NQvD basis sets — have been obtained from the EMSL basis set library [127]. Supplied basis sets include the STO-nG and Pople style basis sets, Dunning's correlation-consistent basis sets, Ahlrich's basis sets

and Huzinaga basis sets. In a very few cases we have corrected the files as obtained from EMSL, however we take no responsibility for any errors inherent in these files.

The ANO and Sadlej-pVTZ polarization basis sets have been obtained from the MOLCAS homepage without any further processing, and should therefore be free of errors. The NQvD has been provided to us by Knut Fægri, and has been slightly reformatted for more convenient processing of the file, hopefully without having introduced any errors.

Below we give a comprehensive list of all basis sets included in the basis set library, together with a list of the elements supported, and the complete reference to be cited when employing a given basis set in a calculation.

Basis set name	Elements	References
***STO*-*n*G**		
STO-2G	H–Ca, Sr	[**?**, **?**]
STO-3G	H–Cd	[**?**, **?**, **?**, **?**]
STO-6G	H–Ar	[**?**, **?**]
Pople-style basis sets		
3-21G	H–Cs	[**?**, **?**, **?**, **?**, **?**, **?**]
3-21G*	H–Cl	3-21G, with polariz. functions from [**?**].
		Note: Polariz. functions only on Na–Cl
3-21++G	H, Li–Ar	3-21G, with diffuse functions from [**?**].
3-21++G*	H–Cl	3-21++G and 3-21G*
4-31G	H–Ar	[**?**, **?**] He,Ne from Gaussian 90
6-31G	H–Ar, Zn	[**?**, **?**, **?**, **?**] He,Ne from Gaussian 90
6-31G*	H–Ar	6-31G, with polariz. functions from [**?**, **?**]
6-31G**	H–Ar	6-31G*, with polariz. functions from [**?**]
6-31+G	H–Ar	6-31G, with diffuse functions from [**?**].
6-31++G	H–Ar	6-31G, with diffuse functions from [**?**].
6-31+G*	H–Ar	6-31+G and 6-31G*
6-31++G*	H–Ar	6-31++G and 6-31G*
6-31++G**	H–Ar	6-31++G and 6-31G**
6-31G(3df,3pd)	H–Ar	6-31G, with polariz. functions from [**?**].
6-311G	H–Ar, Br, I	[**?**, **?**, **?**, **?**]
6-311G*	H–Ar, Br, I	6-311G, with polariz. functions from [**?**, **?**].
6-311G**	H–Ar, Br, I	6-311G, with polariz. functions from [**?**, **?**].
6-311+G*	H–Ne	6-311G* with diffuse functions from [**?**].
6-311++G**	H–Ne	6-311G** with diffuse functions from [**?**].
6-311G(2df,2pd)	H–Ne	6-311G, with polariz. functions from [**?**].
6-311++G(2d,2p)	H–Ne	6-311++G, with polariz. functions from [**?**].
6-311++G(3df,3pd)	H–Ar	6-311++G, with polariz. functions from [**?**].

Basis set name	Elements	References
Huckel	H–Cd	
MINI(Huzinaga)	H–Ca	[?]
MINI(Scaled)	H–Ca	[?, ?]
Dunning-Hay basis sets		
SV(Dunning-Hay)	H, Li–Ne	[?]
SVP(Dunning-Hay)	H, Li–Ne	SV, with polariz. functions from [?, ?].
SVP+Diffuse(Dunning-Hay)	H, Li–Ne	SVP with diffuse functions from [?, ?].
SV+Rydberg(Dunning-Hay)	H, Li–Ne	SV, with polariz. functions from [?].
SV+DoubleRydberg(Dunning-Hay)	H, Li–Ne	SV, with polariz. functions from [?].
DZ(Dunning)	H, B–Ne, Al–Cl	[?, ?]
DZP(Dunning)	H, B–Ne, Al–Cl	DZ, with polariz. functions from [?, ?].
DZP+Diffuse(Dunning)	H, B–Ne	DZP with diffuse functions from [?, ?].
DZ+Rydberg(Dunning)	H, B–Ne, Al–Cl	DZ, with polariz. functions from [?].
DZP+Rydberg(Dunning)	H, B–Ne, Al–Cl	DZP, with polariz. functions from [?].
TZ(Dunning)	H, Li–Ne	[?]
Dunning's correlation consistent basis sets		
Note: H,He (valence only) are included in all core-valence basis sets for convenience.		
cc-pVXZ (X = D,T,Q,5,6)	H–Ne, Al–Ar, Ca, Ga–Kr	[65, 68, 67, ?, ?] *Note*: 6Z only includes H,C–O
cc-pCVXZ (X = D,T,Q,5)	H, He, B–Ne, Na–Ar	cc-pVXZ, with core functions from [65, ?, ?]. *Note*: 5Z only includes H, He, B–Ne.
cc-pwCVXZ (X = D,T,Q,5)	H, He, C–Ne, Al–Ar	cc-pCVXZ with core functions from [65, ?]
aug-cc-pVXZ (X = D,T,Q,5)	H, He, B–Ne, Al–Ar, Ga–Kr	cc-pVXZ, with aug. functions from [65, 66, 67, 68].
aug-cc-pV6Z	H, He, B–Ne, Al–Ar	[?, ?] with aug. functions from [?, ?, ?].
aug-cc-pCVXZ (X = D,T,Q,5)	H, He, B–F, Ne, Al-Ar	aug-cc-pVXZ, cc-pCVXZ and [?, ?]. *Note*: Ne only available for TZ and QZ
n-aug-cc-pVXZ (*n* = d,t,q)	as aug-cc-pVXZ	aug-cc-pVXZ. See Sec. 23.4
n-aug-cc-pCVXZ (*n* = d,t,q)	as aug-cc-pCVXZ	aug-cc-pCVXZ. See Sec. 23.4
cc-pVXZdenfit (X=T,Q,5)	H, B–F, Al–Cl	Turbomole program
cc-pVXZ-DK (X = D,T,Q,5)	H,He,B–Ne, Al–Ar, Ga–Kr	[65, 68, 67, ?] cc-pVXZ re-contracted for Douglas-Kroll calculations.

Basis set name	Elements	References
Ahlrich's VXZ basis sets		
Ahlrichs-VXZ (X = D,T)	H–Kr	[71]
Ahlrichs-TZV	Li–Kr	[72]
Ahlrichs-pVDZ	H–Kr	[71] polariz. functions largely unpublished.
Huzinaga basis sets		
Huz-II	H, C–F, P, S	[**?, ?, ?**] All the Huz basis sets are of
Huz-IIsu2	H, C–F, P, S	[**?, ?, ?**] approximate valence TZ quality
Huz-III	H, C–F, P, S	[**?, ?, ?**]
Huz-IIIsu3	H, C–F, P, S	[**?, ?, ?**]
Huz-IV	H, C–F, P, S	[**?, ?, ?**]
Huz-IVsu4	H, C–F, P, S	[**?, ?, ?**]
GAMESS-VTZ	H, Be–Ne, Na–Ar	[**?, ?, ?**] From GAMESS program
GAMESS-PVTZ	H, Be–Ne	[**?, ?, ?**] From GAMESS program
McLean-Chandler-VTZ	Na–Ar	[**?**]
Wachtersa+f	Sc–Cu	[**?, ?**] with f functions from [**?**]
Sadlej-pVTZ	H–I	[**?, ?, ?, ?**]
		Note: No rare-gas, B, Al, Ga or In basis sets
Sadlej-pVTZ-J	H, C–O, S	[**?**] Sadlej-pVTZ optimized for NMR calcs.
aug-cc-pVTZ-J	H, C–F, S	[**?**] aug-cc-pVTZ optimized for NMR calcs.
ANO basis sets – see Sec. 23.4.		
NQvD		[128]
Almlof-Taylor-ANO	H–Ne	[**?**] Almlof and Taylor ANO
NASA-Ames-ANO	H, B–Ne, Al, P, Ti, Fe, Ni	[**?, ?**]
ano–1	H–Ne	[18] Roos Augmented ANO basis sets
ano–2	Na–Ar	[19]
ano–3	Sc–Zn	[**?**]
ano–4	H–Kr	[**?**] Roos ANO basis sets
raf-r	O, Y–Pd, Hf–Tl, Po, Th, U	Wahlgren/Faegri Relavistic Basis Set
ecp-sdd-DZ	Li–Bi	Stuttgart ECPs (some elements excluded)
ecp-sdd-pseudo	H–Rn	See BASIS/ecp-sdd-pseudo file for references

Chapter 24

Molecular wave functions, SIRIUS

24.1   General notes for the SIRIUS input reference manual

SIRIUS is the part of the code that computes the wave function/density.

The following sections contain a list of all generally relevant keywords to SIRIUS, only currently inactive keywords and some special debug options are omitted.

1. The input for the wave function section must begin with

   ```
   **WAVE FUNCTIONS
   ```
   with no leading blanks. The preceding lines in the input file may contain arbitrary information.

2. Input is directed by keywords written in upper case. Only the first 7 characters including the prompt are significant. The keywords are divided in a number of main input groups. Each main input group is initiated by a "*". For example

   ```
   *ORBITAL INPUT
   ```
   marks the beginning of the input group for orbital input.

3. The keywords belonging to one of the main input groups begin with the prompt ".".

4. Keywords that are necessary to specify are marked by "Required". For other keywords the default values can be used in ordinary runs.

5. Any keyword line beginning with a "!" or "#" will be treated as a comment line. An illegal keyword will cause a dump of all keywords for the current input section.

6. A dump of keywords can be obtained in any input section by specifying the keyword ".OPTIONS". For example, the input

```
**WAVE FUNCTIONS
.OPTIONS
**END OF DALTON INPUT
```
will cause a dump of the labels for the main input groups in SIRIUS, while
```
**WAVE FUNCTIONS
*ORBITAL INPUT
.OPTIONS
**END OF DALTON INPUT
```
will cause a dump of the labels for the "*ORBITAL INPUT" input group in SIRIUS.

7. The SIRIUS input is finished with a line beginning with two stars, *e.g.*

```
**END OF DALTON INPUT
```

24.2 Main input groups in the **WAVE FUNCTIONS input module

The main input groups (those with the "*" prompt) are listed here and the full descriptions are given in the designated sections.

The first input group is always required in order to specify the type of calculation, and follows immediately after the `**WAVE FUNCTIONS` keyword.

The remaining input groups may be specified in any order. In this chapter they are grouped alphabetically, although the short presentation below gather them according to purpose.

The following two input groups are used to modify the molecular environment by adding field-dependent terms in the Hamiltonian and by invoking the self-consistent reaction field model for solvent effects, respectively:

Section 24.2.8 `*HAMILTONIAN`

Section 24.2.16 `*SOLVENT`

The next input group specifies the configurations included in the MCSCF and CI wave functions:

Section 24.2.5 `*CONFIGURATION INPUT`

The two next groups are used to specify initial orbitals and initial guess for the CI vector:

Section 24.2.12 `*ORBITAL INPUT`

Section 24.2.4 `*CI VECTOR`

The two following input groups control the second-order MCSCF optimization:

Section 24.2.11 `*OPTIMIZATION`

Section 24.2.17 `*STEP CONTROL`

The next five groups have special input only relevant for the respective calculation types:

Section 24.2.15 `*SCF INPUT`

Section 24.2.6 `*DFT INPUT`
Section 24.2.9 `*MP2 INPUT`
Section 24.2.10 `*NEVPT2 INPUT`
Section 24.2.3 `*CI INPUT`

The next section is used to select some types of analysis of the final Hartree–Fock, DFT, MCSCF, or CI wave function:

Section 24.2.13 `*POPULATION ANALYSIS`

The next section is used to change the default integral transformation and specify any final integral transformation after convergence (a program following SIRIUS may need a higher transformation level):

Section 24.2.18 `*TRANSFORMATION`

The next two input groups control the amount of printed output and collect options not fitting in any of the other groups:

Section 24.2.14 `*PRINT LEVELS`
Section 24.2.2 `*AUXILIARY INPUT`

Finally we note that there is an input module controlling the calculation of coupled cluster wave functions. This is treated in a separate chapter:

Chapter 27 `*CC INPUT`

The wave function input is finished when a line is encountered beginning with two stars, for example

        `**END OF DALTON INPUT`

or

        `**MOLORB`
        `... formatted molecular orbitals coefficients`
        `**END OF DALTON INPUT`

The `**MOLORB` keyword or the `**NATORB` keyword must be somewhere on the input file and be followed by molecular orbital coefficients if the option for formatted input of molecular orbitals has been specified. Apart from this requirement, arbitrary information can be written to the following lines of the input file.

24.2.1 `**WAVE FUNCTIONS`

Purpose:

Specification of which wave function calculation is to be performed.

Primary keywords, listed in the order the corresponding modules will be executed by the program (if the keyword is set):

`.HF` Restricted closed-shell or one open-shell Hartree–Fock calculation. The occupied orbitals, optimization control etc. are specified in the "*SCF INPUT" submodule. Note: you cannot specify both ".HF" and ".DFT" keywords.

.DFT READ (LUINP,'(A80)) LINE
  Restricted closed-shell, one open-shell or high-spin spin-restricted Kohn-Sham density functional theory calculation. On the following line you must specify which functional to use. The occupied orbitals, optimization control etc. are specified in the "*SCF INPUT" submodule shared with the ".HF" option. The DFT specific input options are collected in the "*DFT INPUT" input submodule. Note: you cannot specify both ".HF" and ".DFT" keywords.

.MP2 Møller-Plesset second-order perturbation theory calculation. Requires ".HF" or previously calculated canonical Hartree–Fock orbitals.

(".FC MVO" in "*SCF INPUT" Calculation of modified virtual SCF orbitals based on the potential determined by the keyword (see comments below). The occupied SCF orbitals are not modified. Note that this keyword is not located in this module but in the "*SCF INPUT" submodule. It is mentioned here to make clear at what point this transformation will be performed, if requested.

.CI Configuration interaction calculation.

.MCSCF Multiconfigurational self-consistent field (MCSCF) calculation.

.NEVPT2 Multireference second-order perturbation theory calculation.

.CC Coupled-cluster calculation. Automatically activates Hartree–Fock (".HF"). After the Hartree–Fock calculation, the CC program is called to do a coupled cluster (response) calculation. For further input options for the CC program see Section 27.1.

.CC ONLY Skip the calculation of a Hartree–Fock wave function, and start directly in the coupled cluster part. Convenient for restarts in the coupled cluster module.

Secondary keywords (in alphabetical order):

.FLAGS READ (LUINP,NMLSIR)
  Read namelist $NMLSIR ... $END
  Only for debugging. Set internal control flags directly. Usage is not documented.

.INTERFACE Write the "SIRIFC" interface file for post-processing programs.

.PRINT READ (LUINP,*) IPRSIR
  General SIRIUS print level and default for all other print parameters in this module.

.RESTART restart SIRIUS, the SIRIUS restart file (SIRIUS.RST) must be available

.STOP READ (LUINP,'(A20)') REWORD

Terminate SIRIUS according to the instruction given on the following line. Three stop points are defined:

1.   " AFTER INPUT"
2.   " AFTER MO-ORTHONORMALIZATION"
3.   " AFTER GRADIENT" (only for MCSCF and 2nd-order HF or DFT)

.TITLE READ (LUINP,'(A)') TITLE(NTIT) Any number of title lines (until next line beginning with "." or "*" prompt). Up to 6 title lines will be saved and used in the output, additional lines will be discarded.

.WESTA Write the "SIRIFC" interface file for the WESTA post-processing program.

24.2.2 *AUXILIARY INPUT

Purpose:

Input which does not naturally fit into any of the other categories.

.NOSUPMAT Do not use P-SUPERMATRIX integrals, but calculate Fock matrices from AO integrals (slower, but requires less disk space). The default is to use the supermatrix file if it exists. See option **.NOSUP** in Chapter 22.1.

24.2.3 *CI INPUT

Purpose:

Options for a CI calculation.

.CIDENSITY Calculate CI one-electron density matrix and natural orbital occupations after convergence.

.CINO Generate CI natural orbitals for CI root number ISTACI, clear the SIRIUS.RST file and write the orbitals with label "NEWORB ". The ".STATE" option must be specified.

.CIROOTS Default: NROOCI = 1
READ (LUINP,*) NROOCI
Converge the lowest NROOCI CI roots to threshold.

.DISKH2 Active two-electron MO integrals on disk (see comments below).

`.MAX ITERATIONS` READ (LUINP,*) MXCIMA

  Max iterations in iterative diagonalization of CI matrix (default = 20).

`.STATE` Default: not specified

  READ (LUINP,*) ISTACI

  Alternative to ".`CIROOTS`". Converge root number ISTACI to threshold, converge all lower roots only to THQMIN (from the "`*OPTIMIZATION`" input group, see p. 215).

`.THRESH` Default = 1.D-05

  READ (LUINP,*) THRCI

  Threshold for CI energy gradient. The CI energy will be converged to approximately the square of this number.

`.THRPWF` Default = 0.05D0 for electronic ground states, and 0.10D0 for excited states

  READ (LUINP,*) THRPWF

  Only CI coefficients greater than threshold are printed (PWF: print wave function).

`.WEIGHTED RESIDUALS` Use energy weighted residuals (see comments below).

`.ZEROELEMENTS` Zero small elements in CI trial vectors (see comments below).

Comments:

  DISKH2: By default the CI module will attempt to place the two-electron integrals with four active indices in memory for more efficient calculation of CI sigma vectors, if memory is insufficient for this the integrals will automatically be placed on disk. The DISKH2 keyword forces the integrals always to be on disk.

  WEIGHTED RESIDUALS: Normally the CI states will be converged to a residual less than the specified threshold, and this will give approximately the squared number of decimal places in the energy. Depending on the value of the energy, the eigenvectors will be converged to different accuracy. If the eigenvectors are wanted with, for instance at least 6 decimal places for property calculations, specify a threshold of 1.0D-6 and weighted residuals.

  ZEROELEMENTS: an experimental option that might save time (if the CI module can use sparseness) by zeroing all elements less than 1.0D-3 times the largest element in the CI trial vector before orthonormalization against previous trial vectors. See also ".SYM CHECK" under "*OPTIMIZATION" (p. 215).

24.2.4 `*CI VECTOR`

Purpose:

  To obtain initial guess for CI vector(s).

.PLUS COMBINATIONS Use with ".STARTHDIAGONAL" to choose plus combination of degenerate diagonal elements (**STRONGLY RECOMMENDED** for calculation of singlet states with ".DETERMINANTS").

.SELECT READ (LUINP,*) ICONF

Select CSF (or determinant if ".DETERMINANTS") no. ICONF as start configuration.

.STARTHDIAGONAL Select configurations corresponding to the lowest diagonal elements in the configuration part of the Hessian (this is the default option).

.STARTOLDCI Start from old CI-vector stored saved on the SIRIUS.RST file.

24.2.5 *CONFIGURATION INPUT

Purpose:

To specify the configuration part in MCSCF and CI calculations.

.CAS SPACE READ (LUINP,*) (NASH(I),I=1,NSYM)

CAS calculation: Active orbitals in each symmetry.

.ELECTRONS Required.

READ (LUINP,*) NACTEL

Number of active electrons (the number of electrons to be distributed in the active orbitals). The total number of electrons is this number plus two times the total number of inactive orbitals.

.INACTIVE ORBITALS Required.

READ (LUINP,*) (NISH(I),I=1,NSYM)

Number of inactive orbitals each symmetry.

.RAS1 ELECTRONS READ (LUINP,*) NEL1MN,NEL1MX

Minimum and maximum number of RAS1 electrons; this can alternatively be specified with ".RAS1 HOLES"

.RAS1 HOLES READ (LUINP,*) NHL1MN,NHL1MX

Minimum and maximum number of holes in RAS1; alternative to ".RAS1 ELECTRONS"

.RAS1 SPACE READ (LUINP,*) (NAS1(I),I=1,NSYM)

RAS calculation: RAS1 orbital space

.RAS2 SPACE READ (LUINP,*) (NAS2(I),I=1,NSYM)

RAS calculation: RAS2 orbital space

`.RAS3 ELECTRONS` READ (LUINP,*) NEL3MN, NEL3MX
    Minimum and maximum number of RAS3 electrons

`.RAS3 SPACE` READ (LUINP,*) (NAS3(I),I=1,NSYM)
    RAS calculation: RAS3 orbital space

`.SPIN MULTIPLICITY` Required for MCSCF and CI wave functions.
    READ (LUINP,*) ISPIN
    For CSF basis: state spin multiplicity $= 2S+1$, where S is the spin quantum number.
    For determinant basis this option determines the minimum spin multiplicity. The M_S
    value is determined as (ISPIN-1)/2.

`.SYMMETRY` Required for MCSCF and CI wave functions.
    READ (LUINP,*) LSYM
    Spatial symmetry of CI and/or MCSCF wave function

Comments:
SYMMETRY Specifies total spatial symmetry of the wave function in D_{2h} symmetry or one
of its subgroups: C_{2v}, C_2h, D_2, C_s, C_i, C_2, C_1. The symmetry number of wave function
follows MOLECULE output ordering of symmetries (D_{2h} subgroup irreps).
CAS and RAS are exclusive and both cannot be specified in the same MCSCF or CI
calculation. One of them *must* be specified.

24.2.6   `*DFT INPUT`

Purpose:
    To specify the parameters of the DFT integration.

`.DFTELS` READ (LUINP,*) DFTELS
    safety threshold – stop if the charge integration gives too large error.

`.DFTTHR` READ (LUINP,*) DFTHRO, DFTHRI
    Thresholds determining accuracy of the numerical integration. The first number de-
    termines the density threshold (contributions to a property from places where the
    density is below the threshold will be skipped) and the second one – orbital threshold
    (orbitals are assumed to be exactly 0 if they are below the threshold). The default
    value for DFTHR0 is $1.0D - 9$ and for DFTRHI is $1.0D - 13$.

`.RADINT` READ (LUINP,*) RADINT

    Determines the quality of the radial part of the grid and corresponds to the upper
    limit of the error in case of an integration on an atom. Default value is $1.0D - 13$.

.ANGINT READ (LUINP,*) ANGINT

Determines the quality of the angular Lebedev grid – the angular integration of spherical harmonics will be exact up to the specified order. Default value is 35.

.GRID TYPE READ (LUINP,*) LINE

Allows specification of different partitioning methods and radial schemes. `BECKE` is Becke partitioning scheme with correction for atomic sizes using Bragg sizes, `BECKEORIG` is the same Becke partitioning scheme but without correction. `SSF` is a partitioning scheme for large molecules designed to reduce the grid generation time. `LMG` select LMG radial scheme adjusted to currently used basis set. Gauss-Chebychev radial scheme of second order is provided for reference and can be selected by keyword `GC2`.

.COARSE Shortcut keyword for radial integration accuracy 10^-11 and angular expansion order equal to 35.

.NORMAL Shortcut keyword for radial integration accuracy 10^-13 and angular expansion order equal to 35.

.FINE Shortcut keyword for radial integration accuracy 10^-13 and angular expansion order equal to 42.

.ULTRAF Shortcut keyword for radial integration accuracy 10^-15 and angular expansion order equal to 65.

24.2.7   DFT functionals

Functionals in DALTON can be divided into two groups: generic and combined functionals. Combined functionals are a linear combination of generic ones. One can always create own combined functionals by using `GGAKey` general functional.

Generic functionals

BECKE Becke exchange correction presented in reference [131]. Observe that full Becke88 exchange functional is sum Slater + Becke.

KT Keal and Tozer GGA exchange functional [132]

LB94 assymptotically correct functional of Leeuwen and Baerends[133]. This functional improves description of the assymptotic density on the expense of core and inner valence.

LYP correlation functional by Lee, Yang and Parr [134, 135].

OPTX exchange functional of Handy [136].

P86 non-local part of the correlation functional of Perdew [137]. PZ81 is usually used for the local part of the functional.

PBEx exchange functional

PW86x Perdew and Wang exchange functional.[138].

PW91c correlation functional of Perdew and Wang. [139]

PZ81 correlation functional of Perdew and Zunger [140].

SLATER Dirac-Slater exchange functional [141, 142, **?**].

VWN3 correlation functional Vosko, Wilk and Nusair (number III) [143].

VWN5 correlation functional Vosko, Wilk and Nusair (number V – the recommended one) [143].

Combined functionals

GGAKey it is an universal keyword allowing to manually construct arbitrary linear combinations of other functionals. Even fractional Hartree–Fock exchange can be specified. This keyword is to be followed by a string of functionals with associated weights. The syntax is `NAME=WEIGHT ....` Example:

```
.DFT
 GGAKey hf=0.2 slater=0.8 becke=0.72 lyp=0.81 vwn3=0.19
```

BLYP is a sum of Slater functional, Becke88 correction and LYP correlation functional. It is equivalent to
`GGAKey slater=1 becke=1 lyp=1`

B3LYP hybrid functional [144] equivalent to:
`GGAKey hf=0.2 slater=0.8 becke=0.72 lyp=0.81 vwn=0.19 .`

B3LYP-G hybrid functional with VWN3 form used for correlation—this is the form used by the Gaussian quantum chemistry program. This functional can be explicitely set up by
`GGAKey hf=0.2 slater=0.8 becke=0.72 lyp=0.81 vwn3=0.19`

BP86 Becke88 exchange functional and Perdew86 correlation functional. The explicit form is:
`GGAKey slater=1 becke=1 pz81=1 p86c=1`

B3P86 variant of `B3LYP` with VWN3 used for local correlation and P86 for the nonlocal part.
    `GGAKey hf=0.2 slater=0.8 becke=0.72 p86c=0.81 vwn3=1`

CAMB3LYP Coulomb Attenuated Method Functional of Yanai, Tew and Handy[145]. This functional accepts additional arguments `alpha`, `beta` and `mu` to modify the fraction of HF exchange for short-range interactions, additional fraction of HF exchange for long-range interaction and the interaction switching factor μ. This input can be specified as follows:

    ```
    .DFT
     CAMB3LYP alpha=0.190 beta=0.460 mu=0.330
    ```

KT1 Slater-VWN5 functional with the KT GGA correction [132, 146].
    `GGAKey slater=1 vwn=1 kt=-0.006`

KT2 differs from KT1 only in that the weights of the Slater and VWN5 functionals are from an empirical fit [132, 146].
    `GGAKey slater=1.07173 vwn=0.576727 kt=-0.006`

KT3 a hybrid functional of Slater, OPTX and KT exhange with the LYP correlation functional [147]. The explicit form is
    `GGAKey slater=1.092 kt=-0.004 lyp=0.864409 optx=-0.925452`

OLYP is the sum of the OPTX exchange functional with the LYP correlation functional [136].
    `GGAKey slater=1.05151 optx=-1.43169 lyp=1`
    Mixed functionals such as OP86 and OPW91 may be constructed with `GGAKey`.

PBE0 a hybrid functional of Perdew, Burke and Ernzerhof with 0.2 weight of exact exchange, 0.75 of `PBEx` functional and `PBEC` correlation functional [148].
    `GGAKey hf=0.25 pbex=0.75 pbec=1`

PBE same as above but with exchange estimated exclusively by `PBEx` functional [149]. This is the form used by eg. CADPAC and NWChem quantum chemistry programs.
    `GGAKey pbex=1 pbec=1`
    Observe that quantum chemistry program Molpro uses `PW91c` correlation functional instead of `PBEC` which is equivalent to following:
    `GGAKey pbex=1 PW91c=1` .

BPW91 is a sum of Slater functional, Becke correction and PW91 correlation functional.
    `GGAKey slater=1 becke=1 pw91c=1`

24.2.8 *HAMILTONIAN

Purpose:

> Add extra terms to the Hamiltonian (for finite field calculations).

.FIELD TERM Default = no fields.

> `READ (LUINP,*) EFIELD(NFIELD)`
>
> `READ (LUINP,*) LFIELD(NFIELD)`
>
> Enter field strength (in atomic units) and property label on separate lines where label is a MOLECULE-style property label on file AOPROPER produced by the property program module, see Chapter 22. The calculation of the necessary property integral(s) must be requested in the "**INTEGRALS" input module.
>
> NOTE: This keyword may be repeated several times for adding more than one finite field (max 20 fields).

.PRINT Default = 0.

> `READ (LUINP,*) IPRH1`
>
> If greater than zero: print the one-electron Hamiltonian matrix, including specified field-dependent terms, in AO basis.

Comments:

> Up to 20 simultaneous fields may be specified by repeating the ".FIELD TERM" keyword. The field integrals are read from file `AOPROPER` with the specified label.

24.2.9 *MP2 INPUT

Purpose:

> To direct MP2 calculation. Note that MP2 energies as well as properties also are available through the coupled cluster module, see Chapter 27.

.MP2 FROZEN Default = no frozen orbitals

> `READ (LUINP,*) (NFRMP2(I),I=1,NSYM)`
>
> Orbitals frozen in MP2 calculation

.PRINT `READ (LUINP,*) IPRMP2`

> Print level for MP2 calculation

Comments:

> The MP2 module expects canonical Hartree–Fock orbitals. The MP2 module will check the orbitals and exits if the Fock matrix has off-diagonal non-negligible elements. If starting from saved canonical Hartree–Fock orbitals from a previous calculations, although

no Hartree–Fock calculation will be done the number of occupied Hartree–Fock orbitals in each symmetry must anyway be specified with the ".DOUBLY OCCUPIED" under "*SCF INPUT".

The MP2 calculation will produce the MP2 energy and the natural orbitals for the density matrix through second order. The primary purpose of this option is to generate good starting orbitals for CI or MCSCF wave functions, but it may of course also be used to obtain the MP2 energy, perhaps with frozen core orbitals. *For valence MCSCF calculations it is recommended that the ".*MP2 FROZEN*" option is used in order to obtain the appropriate correlating orbitals as start for an MCSCF calculation.* As the commonly used basis sets do not contain correlating orbitals for the core orbitals and as the core correlation energy therefore becomes arbitrary, the ".MP2 FROZEN" option can also be of benefit in MP2 energy calculations.

24.2.10   *NEVPT2 INPUT

Purpose:

Calculation of the second order correction to the energy for a CAS–SCF or CAS–CI zero order wavefunction. The user is referred to Chapter 19 on page 129 for a brief introduction to the n–electron valence state second order perturbation theory (NEVPT2).

.THRESH Default = 0.0D0
READ (LUINP,*) THRNEVPT
Threshold to discard small coefficients in the CAS wavefunction

.FROZEN Default = no frozen orbitals
READ (LUINP,*) (NFRNEVPT2(I),I=1,NSYM)
Orbitals frozen in NEVPT2 calculation

.STATE No default provided
READ (LUINP,*) ISTNEVCI
Root number in a CASCI calculation. This keyword is unnecessary (ignored) in the CASSCF case.

Comments:

The use of canonical orbitals for the core and virtual orbitals is strongly recommended since this choice guarantees compliance of the results with a totally invariant form of NEVPT2 (see page 129).

At present the NEVPT2 module can deal with active spaces of dimension not higher than 14.

24.2.11 *OPTIMIZATION

Purpose:

To change defaults for optimization of an MCSCF wave function. Some of the options also affect a QC-HF optimization.

.ABSORPTION READ (LUINP,'(A8)') RWORD

RWORD = ' LEVEL 1', ' LEVEL 2', or ' LEVEL 3'

Orbital absorption in MCSCF optimization at level 1, 2, or 3, as specified (normally level 3, see comments below). This keyword may be repeated to specify more than one absorption level, the program will then begin with the lowest level requested and, when that level is converged, disable the lower level and shift to the next level.

.ACTROT include specified active-active rotations

```
READ (LUINP,*) NWOPT
DO I = 1,NWOPT
  READ (LUINP,*) JWOP(1,I),JWOP(2,I)
END DO
```

JWOP(1:2,I) denotes normal molecular orbital numbers (not the active orbital numbers).

.ALWAYS ABSORPTION Absorption in all MCSCF macro iterations (default is to disable absorption in local region or after ".MAXABS" macro iterations, whichever comes first). Absorption is always disabled after Newton-Raphson algorithm has been used, and thus also when doing ".CORERELAX", because absorption may cause variational collapse if the desired state is excited.

.CI PHP MATRIX Default : MAXPHP = 1 (Davidson's algorithm)

READ (LUINP,*) MAXPHP

PHP is a subblock of the CI matrix which is calculated explicitly in order to obtain improved CI trial vectors compared to the straight Davidson algorithm. The configurations corresponding to the lowest diagonal elements are selected, unless ".PHPRESIDUAL" is specified. MAXPHP is the maximum dimension of PHP, the actual dimension will be less if MAXPHP will split degenerate configurations.

.COREHOLE READ (LUINP,*) JCHSYM,JCHORB

JCHSYM = symmetry of core orbital

JCHORB = the orbital in symmetry JCHSYM with a single core hole

Single core hole MCSCF calculation. The calculation must be of RAS type with only

the single core-hole orbital in RAS1, the state specified with ".STATE" is optimized with the core-hole orbital frozen. The specified core hole orbital must be either inactive or the one RAS1 orbital, if it is inactive then it will switch places with the RAS1 orbital and it will not be possible to also specify ".REORDER". If explicit reordering is required you must reorder the core orbital yourself and let JCHORB point to the one RAS1 orbital. Orbital absorption is activated at level 2. See comments below for more information.

.CORERELAX (ignored if ".COREHOLE" isn't also specified)

Optimize state with relaxed core orbital (using Newton-Raphson algorithm, it is not necessary to explicitly specify ".NR ALWAYS"). It is assumed that this calculation follows an optimization with frozen core orbital and that the orbital has already been moved to the RAS1 space (*i.e.*, the specific value of "JCHORB" under ".COREHOLE" is ignored). Any orbital absorption will be ignored.

.DETERMINANTS Use determinant basis instead of CSF basis (see comments).

.EXACTDIAGONAL Default for RAS calculations.

Use the exact orbital Hessian diagonal.

.FOCKDIAGONAL Default for CAS calculations.

Use an approximate orbital Hessian diagonal which only uses Fock contributions.

.FOCKONLY Activate TRACI option (default : program decides).

Modified TRACI option where all orbitals, also active orbitals, are transformed to Fock type orbitals in each iteration.

.FROZEN CORE ORBITALS READ (LUINP,*) (NFRO(I),I=1,NSYM)

Frozen orbitals : Number of inactive (doubly occupied) orbitals to be frozen in each symmetry (the first NFRO(I) in symmetry I) in MCSCF. Active orbitals and specific inactive orbitals can be frozen with ".FREEZE" under ***ORBITAL INPUT**. Frozen orbitals in SCF are specified in the ***SCF INPUT** input module.

.MAX CI READ (LUINP,*) MAXCIT

maximum number of CI iterations before MCSCF (default = 3).

.MAX MACRO ITERATIONS READ (LUINP,*) MAXMAC

maximum number of macro iterations in MCSCF optimization (default = 15).

.MAX MICRO ITERATIONS READ (LUINP,*) MAXJT

maximum number of micro iterations per macro iteration in MCSCF optimization (default = 24).

.MAXABS READ (LUINP,*) MAXABS
> maximum number of macro iterations with absorption (default = 3).

.MAXAPM READ (LUINP,*) MAXAPM
> maximum number orbital absorptions within a macro iteration (APM : Absorptions Per Macro iteration; default = 5)

.NATONLY Activate TRACI option (default : program decides).
> Modified TRACI option where the inactive and secondary orbitals are not touched (these two types of orbitals are already natural orbitals).

.NEO ALWAYS Always norm-extended optimization (never switch to Newton-Raphson). Note: ".NR ALWAYS" and ".CORERELAX" takes precedence over ".NEO ALWAYS".

.NO ABSORPTION Never orbital absorption (default settings removed)

.NO ACTIVE-ACTIVE ROTATIONS No active-active rotations in RAS optimization.

.NOTRACI Disable TRACI option (default : program decides).

.NR ALWAYS Always Newton-Raphson optimization (never NEO optimization). Note: ".NR ALWAYS" takes precedence over ".NEO ALWAYS".

.OLSEN Use Jeppe Olsen's generalization of the Davidson algorithm.

.OPTIMAL ORBITAL TRIAL VECTORS Generate "optimal" orbital trial vectors [21].

.ORB_TRIAL VECTORS Use also orbital trial vectors as start vectors for auxiliary roots in each macro iteration (CI trial vectors are always generated).

.PHPRESIDUAL Select configurations for PHP matrix based on largest residual rather than lowest diagonal elements.

.SIMULTANEOUS ROOTS Default : NROOTS = ISTATE, LROOTS = NROOTS
> READ (LUINP,*) NROOTS, LROOTS
> NROOTS = Number of simultaneous roots in NEO
> LROOTS = Number of simultaneous roots in NEO at start

.STATE Default = 1
> READ (LUINP,*) ISTATE
> Index of MCSCF Hessian at convergence (1 for lowest state, 2 for first excited state, etc. within the spatial symmetry and spin symmetry specified under *CONFIGURATION INPUT).

`.SYM CHECK` Default: ICHECK = 2 when NROOTS > 1, else ICHECK = -1.
READ (LUINP,*) ICHECK
Check symmetry of the LROOTS start CI-vectors and remove those which have wrong symmetry (*e.g.* vectors of delta symmetry in a sigma symmetry calculation).

```
ICHECK < 0  : No symmetry check.
ICHECK = 1  : Remove those vectors which do not have the same
              symmetry as the ISTATE vector, reassign ISTATE.
ICHECK = 2  : Remove those vectors which do not have the same
              symmetry as the lowest state vector before selecting
              the ISTATE vector.
other values: check symmetry, do not remove any CI vectors.
```

The ".SIMULTANEOUS ROOTS" input will automatically be updated if CI vectors are removed.

`.THRCGR` READ (LUINP,*) THRCGR
Threshold for print of CI gradient. Default is 0.1D0.

`.THRESH` Default = 1.0D-05
READ (LUINP,*) THRMC
Convergence threshold for energy gradient in MCSCF optimization. The convergence of the energy will be approximately the square of this number.

`.TRACI` Activate TRACI option (default : program decides).
Active orbitals are transformed to natural orbitals and the CI-vectors are counter-rotated such that the CI states do not change. The inactive and secondary orbitals are transformed to Fock type orbitals (corresponding to canonical orbitals for closed shell Hartree–Fock). For RAS wave functions the active orbitals are only transformed within their own class (RAS1, RAS2, or RAS3) as the wave function is not invariant to orbital rotations between the classes. For RAS, the orbitals are thus not true natural orbitals, the density matrix is only block diagonalized. Use ".IPRCNO" (see p. 224) to control output from this transformation.

Comments:

COREHOLE: Single core-hole calculations are performed as RAS calculations where the opened core orbital is in the RAS1 space. The RAS1 space must therefore contain one and only one orbital when the COREHOLE option is used, and the occupation must be restricted to be exactly one electron. The orbital identified as the core orbital must be either inactive or the one RAS1 orbital, if it is inactive it will switch places with the one RAS1

orbital. The core orbital (now in RAS1) will be frozen in the following optimization. After this calculation has converged, the CORERELAX option may be added and the core orbital will be relaxed. When CORERELAX is specified it is assumed that the calculation was preceded by a frozen core calculation, and that the orbital has already been moved to the RAS1 space. Default corresponds to the main peak, shake-up energies may be obtained by specifying ".STATE" larger than one. Absorption is very beneficial in core hole calculations because of the large orbital relaxation following the opening of the core hole.

ABSORPTION: Absorption level 1 includes occupied - occupied rotations only (including active-active rotations); level 2 adds inactive - secondary rotations and only active - secondary rotations are excluded at this level; and finally level 3 includes all non-redundant rotation for the frozen CI vector. Levels 1 and 2 require the same integral transformation (because the inactive - secondary rotations are performed using the P-supermatrix integrals) and level 1 is therefore usually not used. Level 3 is the normal and full level, but it can be advantageous to activate level 2 together with level 3 if big inactive-active or occupied-occupied rotations are expected.

ORB_TRIAL: Orbital trial vectors as start vectors can be used for excited states and other calculations with more than one simultaneous roots. The orbital start trial vectors are based on the eigenvectors of the NEO matrix in the previous macro iterations. However, they are probably not cost-effective for multiconfiguration calculations where optimal orbital trial vectors are used and they are therefore not used by default.

SYM CHECK: The symmetry check is performed on the matrix element $\langle VEC1 \mid oper \mid VEC2 \rangle$, where "oper" is the CI-diagonal. It is recommended and the default to use ".SYM CHECK" for excited states, including CI vectors of undesired symmetries is a waste of CPU time.

DETERMINANTS: The kernels of the CI sigma routines and density matrix routines are always performed in determinant basis. However, this keyword specifies that the external representation is Slater determinants as well. The default is that the external representation is in CSF basis as described in chapter 8 of MOTECC-90. The external CSF basis is generally to be preferred to be sure that the converged state(s) have pure and correct spin symmetry, and to save disk space. It is recommended to specify ".PLUS COMBINATIONS" under "*CI VECTOR" for calculations on singlet states with determinants, in particular for excited singlet states which often have lower lying triplet states.

24.2.12 *ORBITAL INPUT

Purpose:

To define an initial set of molecular orbitals and to control the use of super symmetry, frozen orbitals, deletion of orbitals, reordering and punching of orbitals.

.5D7F9G Delete unwanted components in Cartesian d, f, and g orbitals. (s in d; p in f; s

and d in g). By default, HERMIT provides atomic integrals in spherical basis, and this option should therefore not be needed.

.AO DELETE READ (LUINP,*) THROVL

Delete MO's based on canonical orthonormalization using eigenvalues and eigenvectors of the AO overlap matrix.

THROVL: limit for basis set numerical linear dependence (eigenvectors with eigenvalue less than THROVL are excluded). Default is $1.0 \cdot 10^{-6}$.

.CMOMAX READ (LUINP,*) CMAXMO

Abort calculation if the absolute value of any initial MO coefficient is greater than CMAXMO (default : CMAXMO = 10^3). Large MO coefficients can cause significant loss of accuracy in the two-electron integral transformation.

.DELETE READ (LUINP,*) (NDEL(I),I = 1,NSYM)

Delete orbitals, *i.e.* number of molecular orbitals in symmetry "I" is number of basis functions in symmetry "I" minus "NDEL(I)".

Only for use with ".MOSTART" options "FORM12" or "FORM18", it cannot be used with "H1DIAG", "EWMO", or "HUCKEL", and the other restart options as "NEWORB" reads this information from file and this will overwrite what ever was specified here.

.FREEZE Default: no frozen orbitals.

```
   READ (LUINP,*) (NNOR(ISYM), ISYM = 1,NSYM)
   DO ISYM = 1,NSYM
     IF (NNOR(ISYM) .GT. 0) THEN
       READ (LUINP,*) (INOROT(I), I = 1,NNOR(ISYM))
       ...
     END IF
   END DO
```

where INOROT = orbital numbers of the orbitals to be frozen (not rotated) in symmetry "ISYM" both in SCF and MCSCF after any reordering (counting from 1 in each symmetry).

Must be specified after all options reducing the number of orbitals. Frozen occupied orbitals in SCF can only be specified in the ***SCF INPUT** input module and frozen inactive orbitals in MCSCF can only be specified in the ***OPTIMIZATION** input module.

.GRAM-SCHMIDT ORTHONORMALIZATION Default.

Gram–Schmidt orthonormalization of input orbitals.

.LOCALIZATION READ (LUINP,*) REWORD

> Specify that the doubly occupied (inactive) orbitals should be localized after SCF or MCSCF is converged. One option for localization of the orbitals is currently available:

> **BOYLOC**　　Use the Boys localization scheme [**?**].

.MOSTART Molecular orbital input

> READ (LUINP,'(1X,A6)') RWORD
> where RWORD is one of the following:

> **FORM12**　　Formatted input (6F12.8) supplied after ****MOLORB** or ****NATORB** keyword. Use also ".DELETE" if orbitals were deleted.

> **FORM18**　　Formatted input (4F18.14) supplied after ****MOLORB** or ****NATORB** keyword. Use also ".DELETE" if orbitals were deleted.

> **EWMO**　　Start orbitals generated by projecting the EWMO Hückel eigenvectors in a good generally contracted ANO basis set onto the present basis set. The EWMO model generally works better than the EHT model. Default inital guess for molecules in which all atoms have a nuclear charge less than or equal to 36. Note: EWMO/HUCKEL is not implemented yet if any element has a charge larger than 36). The start density will thus be close to one generated from atomic densities, but with molecular valence interaction in the EWMO model. This works a lot better than using a minimal basis set for EMWO.

> **HUCKEL**　　Start orbitals generated by projecting the EHT Hückel eigenvectors in a good generally contracted ANO basis set onto the present basis set. Note: EWMO/HUCKEL is not implemented yet if any element has a charge larger than 36. The start density will thus be close to one generated from atomic densities, but with molecular valence interaction in the Hückel model. This works a lot better than using a minimal basis set for Hückel.

> **H1DIAG**　　Start orbitals that diagonalize one-electron Hamiltonian matrix (default for molecules containing elements with a nuclear larger than 36).

> **NEWORB**　　Input from SIRIUS restart file (**SIRIUS.RST** file) with label "NEWORB "

> **OLDORB**　　Input from SIRIUS restart file (**SIRIUS.RST** file) with label "OLDORB "

> **SIRIFC**　　Input from SIRIUS interface file ("SIRIFC")

.PUNCHINPUTORBITALS Punch input orbitals with label ****MOLORB**, Format (4F18.14). These orbitals may *e.g.* be transferred to another computer and read there with ".MOSTART" followed by " FORM18" on next line from this input section.

.PUNCHOUTPUTORBITALS Punch final orbitals with label `**MOLORB`, Format (4F18.14). These orbitals may *e.g.* be transferred to another computer and read there with ".MOSTART" followed by " FORM18" on next line from this input section.

.REORDER Default: no reordering.

```
   READ (LUINP,*) (NREOR(I), I = 1,NSYM)
   DO I = 1,NSYM
      IF (NREOR(I) .GT. 0) THEN
         READ (LUINP,*) (IMONEW(J,I), IMOOLD(J,I), J = 1,NREOR(I))
      END IF
   END DO
   NREOR(I) = number of orbitals to be reordered in symmetry I
   IMONEW(J,I), IMOOLD(J,I) are orbital numbers in symmetry I.


For example if orbitals 1 and 5 in symmetry 1 should change place, specify
.REORDER
 2 0 0 0
 1 5 5 1
```

Reordering of molecular orbitals (see comments).

.SUPSYM Default is NOSUPSYM.

Enforce automatic identification of "super symmetry" (see comments).

.SYMMETRIC ORTHONORMALIZATION Default: Gram-Schmidt orthonormalization

Symmetric orthonormalization of input orbitals.

.THRSSY READ (LUINP,*) THRSSY

Threshold for identification of "super symmetry" and degeneracies among "super symmetries" from matrix elements of the kinetic energy matrix (default: 5.0D-8).

Comments:

If ".SUPSYM" is specified, then SIRIUS automatically identifies "super symmetry", *i.e.* it assigns orbitals to the irreps of the true point group of the molecule which is a "super group" of the Abelian group used in the calculation. Degenerate orbitals will be averaged and the "super symmetry" will be enforced in the orbitals. Note that "super symmetry" can only be used in the RHF, MP2, MCSCF, and RESPONS modules, and should not be invoked if other modules are used, for example, if `**PROPERTIES` (ABACUS) is invoked. Also, it cannot be used in finite field calculations where the field lowers the symmetry. The initial orbitals must be symmetry orbitals, and the super symmetry analysis is performed

on the kinetic energy matrix in this basis. The ".THRSSY" option is used to define when the kinetic energy matrix element between two orbitals is considered to be zero and when two diagonal matrix elements are degenerate. In the first case the orbitals can belong to different irreps of the supergroup and in the second case the two orbitals are considered to be degenerate. The analysis will fail if there are accidental degeneracies in diagonal elements. This can happen if the nuclear geometry deviates slightly from a higher symmetry point group, for example because too few digits has been used in the input of the nuclear geometry. If the program stops because the super symmetry analysis fails with a degeneracy error, you might consider to use more digits in the nuclear coordinates, to change THRSSY, or to disable super symmetry by not using ".SUPSYM". The value of THRSSY should be sufficiently small to avoid accidental degeneracies and sufficiently large to ignore small errors in geometry and numerical round-off errors.

`.REORDER` can for instance be used for linear molecules to interchange undesired delta orbitals among the active orbitals in symmetry 1 with sigma orbitals. Another example is movement of the core orbital to the RAS1 space for core hole calculation. In general, use of this option necessitates a pre-calculation with STOP AFTER MO-ORTHONORMALIZATION and identification of the various orbitals by inspection of the output.

24.2.13 *POPULATION ANALYSIS

Purpose:

To direct population analysis of the wave function. Requires a set of natural orbitals and their occupation.

`.ALL` Do all options.

`.GROSSALL` Do all gross population analysis. Note that this requires that the dipole length integrals are available on the file `AOPROPER`

`.GROSSMO` Do gross MO population analysis.

`.MULLIKEN` Do Mulliken population analysis

`.NETALL` Do all net population analysis.

`.NETMO` Do net MO population analysis.

`.PRINT` Default = 1
      READ (LUINP,*) IPRMUL
      Print level for population analysis.

`.VIRIAL` Do virial analysis.

24.2.14 *PRINT LEVELS

Purpose:

To control the printing of output.

.CANONI Generate canonical/natural orbitals if the wave function has converged.

.IPRAVE READ (LUINP,*) IPRAVE
Sets print level for routines used in "super symmetry" averaging (default = 0).

.IPRCIX READ (LUINP,*) IPRCIX
Sets print level for setup of determinant/CSF index information (default = 0).

.IPRCNO READ (LUINP,*) IPRCNO
Sets print level for ".TRACI" option (default = 1, to print the natural orbital occupations in each iteration set IPRCNO = 1, higher values will give more print).

.IPRDIA READ (LUINP,*) IPRDIA
Sets print level for calculation of CI diagonal (default = 0)

.IPRDNS READ (LUINP,*) IPRDNS
Sets print level for calculation of CI density matrices (default = 0)

.IPRFCK READ (LUINP,*) IPRFCK
Sets print level in the supersymmetry section (default=0).

.IPRKAP READ (LUINP,*) IPRKAP
Sets print level in routines for calculation of optimal orbital trial vectors (default = 0)

.IPRSIG READ (LUINP,*) IPRSIG
Sets print level for calculation of CI sigma vectors (default = 0)

.IPRSOL READ (LUINP,*) IPRSOL
Sets print level in the solvent contribution parts of the calculation (default = 5).

.NOSUMMARY No final summary of calculation.

.PRINTFLAGS Default: flags set by general levels in ".PRINTLEVELS"

```
READ (LUINP,*) NUM6, NUM4
IF (NUM6 .GT. 0) READ (LUINP,*) (NP6PTH(I), I=1,NUM6)
IF (NUM4 .GT. 0) READ (LUINP,*) (NP4PTH(I), I=1,NUM4)
```

Individual print flag settings (debug option).

.PRINTLEVELS Default: IPRI6 = 0 and IPRI4 = 5
READ (LUINP,*) IPRI6,IPRI4
Print levels on units LUW6 and LUW4, respectively.

.THRPWF READ (LUINP,*) THRPWF
Threshold for printout of wave function CI coefficients (default = 0.05).

24.2.15 *SCF INPUT

Purpose:

This section deals with the closed shell, one open shell and high–spin spin-restricted Hartree–Fock cases and Kohn-Sham DFT. The input here will usually only be used if either ".DFT" or ".HF" has been specified under "**WAVE FUNCTIONS" (though it is also needed for MP2 calculations based on saved closed-shell HF orbitals). Single configuration cases with more than one open shell are handled by the general *CONFIGURATION INPUT section, although high-spin restricted open-shell Hartree–Fock or Kohn–Sham DFT calculations can be done using the ".HSROHF" described here.

.AUTOCCUPATION Default for SCF calculations starting from Hückel or H1DIAG starting orbitals.

Allow the distribution of the Hartree–Fock occupation numbers over symmetries to change based on changes in orbital ordering during DIIS optimization. This keyword is incompatible with ".HSROHF" and ".COREHOLE", or if the HF calculation is followed by CI or MCSCF..

.C2DIIS Use Harrell Sellers' C2-DIIS algorithm instead of Pulay's C1-DIIS algorithm (see comments).

.COREHOLE READ (LUINP,*) JCHSYM,JCHORB
JCHSYM = symmetry of core orbital
JCHORB = the orbital in symmetry JCHSYM with a single core hole
Single core hole open shell RHF calculation, ".OPEN SHELL" must not be specified. The specified core orbital must be inactive. The number of doubly occupied orbitals in symmetry JCHSYM will be reduced with one and instead an open shell orbital will be added for the core hole orbital. If the specified core orbital is not the last occupied orbital in symmetry JCHSYM it will switch places with that orbital and user-defined reordering is not possible. If explicit reordering is required you must also reorder the core orbital yourself and let JCHORB point to the last occupied orbital of symmetry JCHSYM. See comments below.

.CORERELAX (ignored if ".COREHOLE" isn't also specified)

Optimize core hole state with relaxed core orbital using Newton-Raphson algorithm. It is assumed that this calculation follows an optimization with frozen core orbital and the specific value of "JCHORB" under ".COREHOLE" is ignored (no reordering will take place).

.DOUBLY OCCUPIED READ (LUINP,*) (NRHF(I),I=1,NSYM)

Explicit specification of number of doubly occupied orbitals in each symmetry for DFT, RHF and MP2 calculations. This keyword is required when Hartree–Fock or MP2 is part of a multistep calculation which includes an MCSCF wave function. Otherwise the program by default will try to guess the occupation, corresponding to the ".AUTOCC" keyword.

.ELECTRONS READ (LUINP,*) NRHFEL

Number of electrons in the molecule. By default, this number will be determined on the basis of the nuclear charges and the total charge of the molecule as specified in the `MOLECULE.INP` file. The keyword is incompatible with the keywords ".DOUBLY OCCUPIED", ".OPEN SHELL", and ".HSROHF".

.FC MVO READ (LUINP,*) (NMVO(I), I = 1,NSYM)

Modified virtual orbitals using Bauschlichers suggestion (see Ref. [150]) for CI or for start guess for MCSCF. The modified virtual orbitals are obtained by diagonalizing the virtual-virtual block of the Fock matrix constructed from NMVO(1:NSYM) doubly occupied orbitals. The occupied SCF orbitals (i.e those specified with ".DOUBLY OCCUPIED" and ".OPEN SHELL" or by automatic occupation) are not modified. The construction of modified virtual orbitals will follow any SCF and MP2 calculations. See comments below.

.FOCK ITERATIONS READ (LUINP,*) MAXFCK

Maximum number of closed-shell Roothaan Fock iterations (default = 0).

.FROZEN CORE ORBITALS READ (LUINP,*) (NFRRHF(I),I=1,NSYM)

Frozen orbitals per symmetry (if MP2 follows then at least these orbitals must be frozen in the MP2 calculation). NOTE: no Roothaan Fock iterations allowed if frozen orbitals.

.H1VIRT Use the virtual orbitals that diagonalize the one-electron Hamiltonian operator.

.HSROHF READ (LUINP,*) (NROHF(I),I=1,NSYM)

High spin restricted open-shell Hartree–Fock. Specify the number of singly occupied orbitals in each irreducible representation of the molecular point group. Only the high-spin state of these singly-occupied orbitals will be made and used in the calculations.

.MAX DIIS ITERATIONS READ (LUINP,*) MXDIIS
Maximum number of DIIS iterations (default = 60).

.MAX ERROR VECTORS READ (LUINP,*) MXEVC
Maximum number of DIIS error vectors (default = 10, if there is sufficient memory available to hold these vectors in memory).

.MAX MACRO ITERATIONS READ (LUINP,*) MXHFMA
Maximum number of QCSCF macro iterations (default = 15).

.MAX MICRO ITERATIONS READ (LUINP,*) MXHFMI
Maximum number of QCSCF micro iterations per macro iteration (default = 12).

.NODIIS Do not use DIIS algorithms (default: use DIIS algorithm).

.NONCANONICAL No transformation to canonical orbitals.

.NOQCSCF No quadratically convergent SCF iterations. Default is to switch to QCSCF if DIIS doesn't converge.

.OPEN SHELL Default = no open shell
READ (LUINP,*) IOPRHF
Symmetry of the open shell in a one open shell calculation. See also ".HSROHF" for high-spin ROHF with more than one singly occupied orbital.

.PRINT READ (LUINP,*) IPRRHF
Resets general print level to **IPRRHF** in Hartree–Fock calculation (if not specified, global print levels will be used).

.SHIFT READ (LUINP,*) SHFTLVL
Initial value of level-shift parameter in DIIS iterations. The default value is 0.0D0 (no level shift). May be tried if convergence problems in DIIS. The value is added to the diagonal of the occupied part of the Fock matrix before Roothaan diagonalization, reducing the mixing of occupied and virtual orbitals (step restriction). NOTE that the value should thus be negative. The DIIS routines will automatically invoke level-shifting (step restriction) if DIIS seems to be stalling.

.THRESH Default = 1.0D-06
READ (LUINP,*) THRRHF
Hartree–Fock convergence threshold for energy gradient. The convergence of the energy will be approximately the square of this number.

Comments:
By default, the RHF/DFT part of a calculation will consist of :

1. MAXFCK Roothaan Fock iterations (early exit if convergence or oscillations). However, the default is that no Roothaan Fock iterations are done unless explicitly requested through the keyword ".FOCK I".

2. MXDIIS DIIS iterations (exit if convergence, *i.e.* gradient norm less than THRRHF, and if convergence rate too slow or even diverging).

3. Unless NOQCSCF, quadratically convergent Hartree–Fock/DFT until gradient norm less than THRRHF.

4. If ".FC MVO" has been specified then the virtual SCF orbitals will be modified by diagonalizing the virtual-virtual block of a modified Fock matrix: the Fock matrix based on the occupied orbitals specified after the keyword, a good choice is the inactive (doubly occupied) orbitals in the following CI or MCSCF. The occupied SCF orbitals will not be modified. If the RHF calculation is followed by a CI or an MCSCF calculation, ".FC MVO" will usually provide much better start orbitals than the canonical orbitals (canonical orbitals will usually put diffuse, non-correlating orbitals in the active space).
   WARNING: if both ".MP2" and ".FC MVO" are specified, then the MP2 orbitals will be destroyed and replaced with ".FC MVO" orbitals.

In general ".DOUBLY OCCUPIED" should be specified for CI or MCSCF wave function calculations – you anyway need to know the distribution of orbitals over symmetries to specify the "*CI INPUT" input. For RHF or MP2 calculations the orbital occupation will be determined on the basis of the nuclear charges and molecular charge of the molecule as specified in the `MOLECULE.INP` file.

By default, starting orbitals and initial orbital occupation will be determined automatically on the basis of a Hückel calculation (for molecules where all nuclear charges are less than or equal to 36), corresponding to the ".AUTOCC" keyword. If problems is experienced due to the Hückel starting guess, it can be avoided by requiring another set of starting orbitals (*e.g.* `H1DIAG`).

It is our experience that it is usually most efficient not to perform any Roothaan Fock iterations before DIIS is activated, therefore, MAXFCK = 0 as default. The algorithm described in Harrell Sellers, Int. J. Quant. Chem. **45**, 31-41 (1993) is also implemented, and may be selected with ".C2DIIS".

FC MVO: This option can be used without a Hartree–Fock calculation to obtain compact virtual orbitals, but ".DOUBLY OCCUPIED" must be specified anyway in order to identify the virtual orbitals to be transformed.

COREHOLE: Enable SCF single core-hole calculations. To perform an SCF core hole calculation just add the ".COREHOLE" keyword to the input for the closed-shell RHF

ground state calculation, specifying from which orbital to remove an electron, and provide the program with the ground state orbitals using the appropriate ".MOSTART" option (normally NEWORB). Note that this is different from the MCSCF version of ".COREHOLE" under "*OPTIMIZATION" (p. 215); in the MCSCF case the user must explicitly move the core hole orbital from the inactive class to RAS1 by modifying the "*CONFIGURATION INPUT" (p. 208) specifications between the initial calculation with filled core orbitals and the core hole calculation. The core hole orbital will be frozen in the following optimization. After this calculation has converged, the CORERELAX option may be added and the core orbital will be relaxed. When CORERELAX is specified it is assumed that the calculation was preceded by a frozen core calculation, and that the orbital has already been moved to the open shell orbital. Only the main peak can be obtained in SCF calculations, for shake-up energies MCSCF must be used.

24.2.16   *SOLVENT

Purpose:

Model solvent effects with the self-consistent reaction field model. Any specification of dielectric constant(s) will activate this model.

.CAVITY Required, no defaults.
READ (LUINP,*) RSOLAV
Enter radius of spherical cavity in atomic units (bohr).

.DIELECTRIC CONSTANT READ (LUINP,*) EPSOL
Enter relevant dielectric constant of solvent.

.INERSINITIAL READ (LUINP,*) EPSOL, EPPN
Enter static and optical dielectric constant of solvent for calculation of the initial state defining inertial polarization. Note that the optical dielectric constant specified here only will be used in case there is a calculation of response properties, for which this is an alternative input to the use of the keywork .INERSFINAL.

.INERSFINAL READ (LUINP,*) EPSTAT,EPSOL
Enter static and optical dielectric constants of solvent for calculation of the final state with inertial polarization from a previous calculation with ".INERSINITIAL". This can for example be used to optimize an excited electronic state with inertial polarization from a previous ground state calculation. This keyword an also be used to specify the static and optical dielectric constants for non-equilibrium solvation linear, quadratic, or cubic response functions, see also Sec. 15.2.2, but this is usually easier done with ".INERSINITIAL" (requires only one DALTON calculation instead of two).

`.MAX L` Required, no defaults.
    READ (LUINP,*) LSOLMX
    Enter maximum L quantum number in multipole expansion of charge distribution in
    cavity.

`.PRINT` READ (LUINP,*) IPRSOL
    Print level in solvent module routines (default = 0).

Comments:
    One and only one of ".DIELECTRIC CONSTANT", ".INERSINITIAL", and ".INERSFINAL"
must be specified.

24.2.17 *STEP CONTROL

Purpose:
    User control of the NEO restricted step optimization.

`.DAMPING FACTOR` Default = 1.0D0
    READ (LUINP,*) BETA
    Initial value of damping (BETA).

`.DECREMENT FACTOR` Default = 0.67D0
    READ (LUINP,*) STPRED
    Decrement factor on trust radius

`.GOOD RATIO` Default = 0.8D0
    READ (LUINP,*) RATGOD
    Threshold ratio for good second order agreement: the trust radius can be increased if
    ratio is better than RATGOD.

`.INCREMENT FACTOR` Default = 1.2D0
    READ (LUINP,*) STPINC
    Increment factor on trust radius.

`.MAX DAMPING` Default = 1.0D6
    READ (LUINP,*) BETMAX
    Maximum damping value.

`.MAX STEP LENGTH` Default = 0.7D0
    READ (LUINP,*) STPMAX
    Maximum acceptable step length, trust radius will never be larger than STPMAX
    even if the ratio is good as defined by GOOD RATIO.

`.MIN DAMPING` Default = 0.2D0
    READ (LUINP,*) BETMIN
    Minimum damping value

`.MIN RATIO` Default = 0.4D0 for ground state, 0.6 for excited states
    READ (LUINP,*) RATMIN
    Threshold ratio for bad second order agreement: the trust radius is to be decreased
    if ratio is worse than RATMIN.

`.NO EXTRA TERMINATION TESTS` Skip extra termination tests and converge micro iterations
    to threshold. Normally the micro iterations are terminated if the reduced NEO matrix
    has more negative eigenvalues than corresponding to the desired state, because then
    we are in a "superglobal" region and we just want to step as quickly as possible to
    the region where the Hessian (and NEO matrix) has the correct structure. Further
    convergence is usually wasted.

`.REJECT THRESHOLD` Default = 0.25 for ground state, 0.4 for excited states
    READ (LUINP,*) RATREJ
    Threshold ratio for unacceptable second order agreement: the step must be rejected
    if ratio is worse than RATREJ.

`.THQKVA` Default: 8.0 for MCSCF; 0.8 for QCSCF
    READ (LUINP,*) THQKVA
    Convergence factor for micro iterations in local (quadratic) region: THQKVA*(norm
    of gradient)**2

`.THQLIN` Default: 0.2D0
    READ (LUINP,*) THQLIN
    Convergence factor for micro iterations in global (linear) region:
    THQLIN*(norm of gradient)

`.THQMIN` Default: 0.1D0
    READ (LUINP,*) THQMIN
    Convergence threshold for auxiliary roots in NEO MCSCF optimization.

`.TIGHT STEP CONTROL` Tight step control also for ground state calculations (tight step
    control is always enforced for excited states)

`.TOLERANCE` Default = 1.1D0
    READ (LUINP,*) RTTOL
    Acceptable tolerance in deviation of actual step from trust radius (the default value
    of 1.1 corresponds to a maximum of 10% deviation).

.TRUST RADIUS Default = STPMAX=0.7D0 or, if restart, trust radius determined by previous iteration.
READ (LUINP,*) RTRUST
Initial trust radius.

24.2.18 *TRANSFORMATION

Purpose:

Transformation of two-electron integrals to molecular orbital basis.

.FINAL LEVEL READ (LUINP,*) ITRFIN
Final integral transformation level (only active if the keyword ".INTERFACE" has been specified, or this is an ABACUS or RESPONSE calculation.

.LEVEL READ (LUINP,*) ITRLVL
Integral transformation level (see comments).

.OLD TRANSFORMATION Use existing transformed integrals

.PRESORT Indicates that the transformed two-electron integrals are already present generated through the keyword .SORT I in the **INTEGRALS** input module. This keyword is true if ".PRESORT" has been set in the general input module.

.PRINT READ (LUINP,*) IPRTRA
Print level in integral transformation module

.RESIDENT MEMORY READ (LUINP,*) MWORK
On virtual memory computers, the transformation will run more efficiently if it can be kept within the possible resident memory size: the real memory size. SIRIUS will attempt to only use MWORK double precision words in the transformation.

Comments:

There are several types of integral transformations which may be specified by the two transformation level keywords.

0: CI calculations, MCSCF gradient (default if CI, but no MCSCF specified). One index all orbitals, three indices only active orbitals.

1: Obsolete, do not use.

2: Default for MCSCF optimization. All integrals needed for SIRIUS second-order MCSCF optimization, including the integrals needed to explicitly construct the diagonal of the orbital Hessian. Two indices occupied orbitals, two indices all orbitals, with some reduction for inactive indices. Both (cd/ab) and (ab/cd) are stored.

3: Same integrals as 2, including also the (ii/aa) and (ia/ia) integrals for exact inactive-secondary diagonal elements of the orbitals Hessian.

4: All integrals with minimum two occupied indices.

5: 3 general indices, one occupied index. Required for MP2 natural orbital analysis (the MP2 module automatically performs an integral transformation of this level).

10: Full transformation.

24.3   **MOLORB** input module

If formatted input of the molecular orbitals has been specified in the *ORBITAL INPUT section, then SIRIUS will attempt to find the two-star label "**MOLORB" in the input file and read the orbital coefficients from the lines following this label.

Chapter 25

HF, SOPPA, and MCSCF molecular properties, ABACUS

25.1 Directives for evaluation of HF, SOPPA, and MCSCF molecular properties

The following directives may be included in the input to ABACUS. They are organized according to the program section (module) names in which they can appear.

25.1.1 General: `**PROPERTIES`

This module controls the main features of the HF, SOPPA, and MCSCF property calculations, that is, which properties is to be calculated. In addition it includes directives affecting the performance of several of the program sections. This includes HF and MCSCF molecular gradients and Hessians. It should be noted, however, that the specification of what kind of walk (minimization, location of transition states, dynamical walks) is given in the `*WALK` or `*OPTIMIZE` submodules in the general input module. See also Chapter 7.

See Chapter 27 for specification of CC property calculations.

Note that RESPONSE (Chapter 26) is the most general part of the code for calculating many different electronic linear, quadratic, or cubic molecular response properties based on SCF, MCSCF, or CI wave functions or Kohn–Sham DFT. Some of these SCF/MCSCF properties can also be requested from the `**PROPERTIES` input modules described here. NOTE: for such properties you should request them either here or in `**RESPONSE`, otherwise you will calculate them twice! Usually the output is nicest here in the `**PROPERTIES` module (*e.g.* collected in tables and in often used units, most properties are only given in atomic units in RESPONSE), and nuclear contributions are included if relevant. No nuclear contributions are added in RESPONSE. Some specific properties, especially those involving nuclear derivatives, can only be calculated via `**PROPERTIES`. Other properties, for example

quadratic and cubic molecular response functions can only be calculated in the `**RESPONSE` module.

`.ALPHA` Invokes the calculation of frequency dependent polarizabilities. Combined with the keyword `.SOPPA` or `.SOPPA(CCSD)` it invokes a SOPPA or SOPPA(CCSD) calculation of the frequency dependent polarizability.

`.CAVORG`

    READ (LUCMD,*) (CAVORG(ICOOR), ICOOR = 1, 3)

Reads the origin to be used for the cavity during a solvent calculation. By default this is chosen to be the center of mass. Should by used with care, as it has to correspond to the center used in the evaluation of the undifferentiated solvent integrals in the HERMIT section, see Chapter 22.2.3.

`.CTOCD` Starts the calculation of the magnetic properties with the CTOCD-DZ method (Ref. [58, 59, 60]). This sets also automatically the .NOLOND option since the CTOCD-DZ formalism is gauge independent for Nuclear Magnetic Shieldings. The default gauge origin is choosen to be the center of mass. `.CTOCD` and `.SOPPA` or `.SOPPA(CCSD)` can be combined to perform CTOCD calculations at the SOPPA or SOPPA(CCSD) level.

`.DIPGRA` Invokes the calculation of dipole moment gradients (commonly known also as Atomic Polar Tensors (APTs)) as described in Ref. [151]. If combined with a request for `.VIBANA` this will generate IR intensities.

`.DIPORG`

    READ (LUCMD, *) (DIPORG(ICOOR), ICOOR = 1, 3)

Reads in a user defined dipole origin. This may affect properties in which changes in the dipole origin is canceled by similar changes in the nuclear part. It should also be used with care, as the same dipole origin must be used during the integral evaluation sections, in particular if one is doing numerical differentiation with respect to electric field perturbations. For such finite-field calculations, we refer to Chapter 14, which deals with finite field calculations. It is primarily used for debugging.

`.ECD` Invokes the calculation of Electronic Circular Dichroism (ECD) as described in Ref. [79, 85]. This necessitates the specification of the number electronic excitations in each symmetry, given in the `*EXCITA` module. The reader is refered to the section where the calculation of ECD is described in more detail (Sec. 11.2).

`.EXCITA` Invokes the calculation of electronic excitation energies as residues of linear response functions as described by Olsen and Jørgensen [48]. It also calculates closely

related properties like transition moments and rotatory strengths. Combined with the keyword `.SOPPA` or `.SOPPA(CCSD)` it invokes a SOPPA or SOPPA(CCSD) calculation of electronic excitation energies and transition moments.

`.EXPFCK` Invokes the simultaneous calculation of two-electron expectation values and derivative Fock-matrices. This is default in direct and parallel runs in order to save memory. In ordinary calculations the total CPU time will increase as a result of invoking this option.

`.EXPGRA` Calculates the gradient of the orbital exponents. This can be used to optimize the so-called polarization consistent basis sets [?] if combined with a suitable script for predicting new orbital exponents based on this gradient.

`.GAUGEO`

```
READ (LUCMD, *) (GAGORG(ICOOR), ICOOR = 1, 3)
```

Reads in a user defined gauge origin and overwrite both the `.NOCMC` option, as well as the default value of center-of-mass coordinates. Note that an unsymmetric position of the gauge origin will lead to wrong results in calculations employing symmetry, as the program will not be able to detect that such a choice of gauge origin breaks the symmetry of the molecule.

`.HELLMA` Tells the program to use the Hellman–Feynman approximation when calculating the molecular gradients and Hessians [152, 153, 154]—that is, all contributions to the molecular gradient and Hessians from differentiation of the orbitals are ignored. Requires large basis sets in order to give reliable results, but does not require any differentiated two-electron integrals.

`.INPTES` Checks the input in the `**PROPERTIES` input section and then stops.

`.LINEAR` Invokes the linear coupling model for estimation of Franck-Condon factors [155]. In this model, the gradient of an excited state is combined with the ground-state vibrational frequencies and normal modes to provide vibronic coupling constants. Requires that the DALTON.HES for the ground electronic state is available, and that the keyword `.VIBANA` also is activated.

`.MAGNET` Invokes the calculation of the molecular magnetizability (commonly known as magnetic susceptibility) as described in Ref. [56] and the rotational g tensor (see keyword `.MOLGFA`). By default this is done using London orbitals in order to ensure fast basis set convergence as shown in Ref. [56]. The use of London orbitals can be disabled by the keyword `.NOLOND`.

Furthermore, the natural connection (Ref. [64, 82]) is the default in order to ensure numerically stable results. The natural connection can be turned off by the keyword `.NODIFC`, in which case the symmetric connection will be used.

The gauge origin is chosen to be the center of mass of the molecule. This origin can be changed by the two keywords `.GAUGEO` and `.NOCMC`. This will of course not affect the total magnetizability, only the magnitude of the dia- and paramagnetic terms.

Combined with the keyword `.SOPPA` or `.SOPPA(CCSD)` it invokes a SOPPA or SOPPA(CCSD) calculation of the magnetizability and the rotational g tensor (Ref. [62]). London orbitals are automatically disabled in SOPPA or SOPPA(CCSD) calculations.

`.MAGNET` in combination with the keyword `.CTOCD` invokes a calculation without the use of London orbitals both with the CTOCD-DZ method (Ref [58, 59]) and with the common origin method. Changing the default value of the gauge origin could give wrong results!

`.MOLGFA` Invokes the calculation of the rotational g tensor as described in Ref. [74] and the molecular magnetizability (see keyword `.MAGNET`). By default this is done using London orbitals and the natural connection. The use of London orbitals can be turned off by the keyword `.NOLOND`.

By definition the gauge origin of the molecular g-factor is to be the center of mass of the molecule, and although the gauge origin can be changed through the keywords `.NOCMC` and `.GAUGEO`, this is not recommended, and may give erroneous results.

Note that if the isotopic constitution of the molecule is such that the vibrational wave function has lower symmetry than the electronic wave function, care must be taken to ensure the symmetry corresponds to the symmetry of the nuclear framework. The automatic symmetry detection routines will in general ensure that this is the case.

Combined with the keyword `.SOPPA` or `.SOPPA(CCSD)` it invokes a SOPPA or SOPPA(CCSD) calculation of the magnetizability and the rotational g tensor (Ref. [62]). London orbitals are automatically disabled in SOPPA or SOPPA(CCSD) calculations.

`.MOLGRA` Invokes the calculation of the analytical molecular gradient as described in Ref. [22].

`.MOLHES` Invokes the calculation of the analytical molecular Hessian and gradient as described in Ref. [22].

`.NMR` Invokes the calculation of both parameters entering the NMR spin-Hamiltonian, that is nuclear shieldings and indirect nuclear spin-spin coupling constants. The reader is refered to the description of the two keywords `.SHIELD` and `.SPIN-S`.

.NOCMC This keyword sets the gauge origin to the origin of the Cartesian Coordinate system, that is (0,0,0). This keyword is automatically invoked in case of VCD calculations.

.NODARW Turns off the calculation of the Darwin correction. By default the two major relativistic corrections to the energy in the Breit-Pauli approximation, the mass-velocity and Darwin corrections, are calculated perturbatively.

.NODIFC Disables the use of the natural connection, and the symmetric connection is used instead. The natural connection and its differences as compared to the symmetric connection is described in Ref. [64, 82].

As the symmetric connection may give numerically inaccurate results, it's use is not recommended for other than comparisons with other programs.

.NOHESS Turns off the calculation of the analytical molecular Hessian. This option overrides any request for the calculation of molecular Hessians.

.NOLOND Turns off the use of London atomic orbitals in the calculation of molecular magnetic properties. The gauge origin is by default then chosen to be the center of mass. This can be altered by the keywords **.NOCMC** and **.GAUGEO**.

.NOMASV Turns off the calculation of the mass-velocity correction. By default the two major relativistic corrections to the energy in the Breit-Pauli approximation, the mass-velocity and Darwin corrections, are calculated perturbatively.

.NQCC Calculates the nuclear quadrupole moment coupling constants.

.NUMHES In VROA or Raman intensity calculations, use the numerical Hessian calculated from the analytical gradients instead of a fully analytical molecular Hessian calculation in the final geometry.

.OPTROT Requests the calculation of the optical rotation of a molecule [42, 80]. By default the optical rotation is calculated both with and without the use of London orbitals. Note that in the formalism used in DALTON, this quantity vanishes in the static limit, and frequencies need to be set in the ***ABALNR** input module. See also description in Chapter 11.3.

.PHASEO
```
READ (LUCMD, *) (ORIGIN(ICOOR), ICOOR = 1, 3)
```
Changes the origin of the phase-factors entering the London atomic orbitals. This will change the value of all of the contributions to the different magnetic field dependent properties when using London atomic orbitals, but the total magnetic properties will remain unchanged. To be used for debugging purposes only.

.POLARI Invokes the calculation of frequency-independent polarizabilities. See the keyword **.ALPHA** in this input section for the calculation of frequency-dependent polarizabilities.

.POPANA Invokes a population analysis based on the dipole gradient as first introduced by Cioslowski [39]. This flag also invokes the **.DIPGRA** flag and the **.POLARI** flags. Note that the charges obtained in this approach is not without conceptual problems (as are the Mulliken charges) [**?**].

.PRINT

    READ (LUCMD, *) IPRDEF

Set default print level for the calculation. Read one more line containing print level. Default print level is the value of IPRDEF from the general input module.

.QUADRU Calculates the molecular quadrupole moment. This includes both the electronic and nuclear contribution to the quadrupole moment. These will printed separately only if a printed level of 2 or higher has been chosen. Note that the quadrupole moment is defined according to Buckingham [43]. The quadrupole moment is printed in the molecular input orientation as well as being transformed to the principal moments of inertia coordinate system.

.RAMAN Calculates Raman intensities, as described in Ref. [42]. This property needs a lot of settings in order to perform correctly, and the reader is therefore refered to Section 11.4, where the calculation of this property is described in more detail.

.REPS

    READ (LUCMD, *) NREPS

    READ (LUCMD, *) (IDOSYM(I),I = 1, NREPS)

Consider perturbations of selected symmetries only. Read one more line specifying how many symmetries, then one line listing the desired symmetries. This option is currently only implemented for geometric perturbations.

.SELECT

    READ (LUCMD,*) NPERT

    READ (LUCMD, *) (IPOINT(I),I=1,NPERT)

Select which nuclear geometric perturbations are to be considered. Read one more line specifying how many perturbations, then on a new line the list of perturbations to be considered. By default, all perturbations are to be considered, but by invoking this keyword, only those perturbations specified in the sequence will be considered.

The perturbation ordering follows the ordering of the symmetrized nuclear coordinates. This ordering can be obtained by setting the print level in the ***READIN** module to 20 or higher.

.SHIELD Invokes the calculation of nuclear shielding constants. By default this is done using London orbitals in order to ensure fast basis set convergence as shown in Ref. [54, 55]. The use of London orbitals can be disabled by the keyword **.NOLOND**.

Furthermore, the natural connection (Ref. [64, 82]) is the default in order to ensure numerically stable results as well as physically interpretable results for the paramagnetic and diamagnetic terms. The natural connection can be turned off by the keyword **.NODIFC** in which case the symmetric connection is used instead.

The gauge origin is chosen to be the center of mass of the molecule. This origin can be changed by the two keywords **.GAUGEO** and **.NOCMC** . This choice of gauge origin will not affect the final shieldings if London orbitals are used, only the size of the dia- and paramagnetic contributions.

Combined with the keyword **.SOPPA** or **.SOPPA(CCSD)** it invokes a SOPPA or SOPPA(CCSD) calculation of the Nuclear Magnetic Shieldings (Ref. [58, 59, 60]). London orbitals are automatically disabled in SOPPA or SOPPA(CCSD) calculations. Gauge origin independent SOPPA or SOPPA(CCSD) calculations of Nuclear Magnetic Shieldings can be carried out with the CTOCD-DZ method (see Refs. [58, 59, 60]) using the keyword **.CTOCD**.

In combination with the keyword **.CTOCD** this invokes a calculation of the Nuclear Magnetic Shieldings without the use of London orbitals but with both the CTOCD-DZ method (Ref. [58, 59, 60]) and with the common origin method. For the CTOCD-DZ method the Nuclear Magnetic Shieldings are given in the output file for both the origin at the center of mass and at the respective atoms. Changing the default value of the gauge origin could give wrong results!

.SOPPA Indicates that the requested molecular properties be calculated using the second-order polarization-propagator approximation [51]. This requires that the MP2 energy and wave function have been calculated. London orbitals can not be used together with the SOPPA approximation.

.SOPPA(CCSD) Indicates that the requested molecular properties be calculated using the Second-Order Polarization-Propagator Approximation with Coupled Cluster Singles and Doubles Amplitudes [53, 63, 52, 60]. This requires that the CCSD energy and wave function have been calculated. London orbitals can not be used together with the SOPPA(CCSD) approximation.

.SPIN-R Invokes the calculation of spin-rotation constants as described in Ref. [74]. By default this is done using London orbitals and the natural connection. The use of London orbitals can be turned off by the keyword **.NOLOND**.

By definition the gauge origin of the spin-rotation constant is to be the center of mass of the molecule, and although the gauge origin can be changed through the keywords `.NOCMC` and `.GAUGEO`, this is not recommended, and may give erroneous results.

In the current implementation, symmetry dependent nuclei cannot be used during the calculation of spin-rotation constants.

`.SPIN-S` Invokes the calculation of indirect nuclear spin-spin coupling constants. By default all spin-spin couplings between nuclei with naturally occuring isotopes with abundance more than 1% and non-zero spin will be calculated, as well as all the different contributions (Fermi contact, dia- and paramagnetic spin-orbit and spin-dipole). The implementation is described in Ref. [61].

As this is a very time consuming property, it is recommended to consult the chapter describing the calculation of NMR-parameters (Ch. 10). The main control of which contributions and which nuclei to calculate spin-spin couplings between is done in the `*SPIN-S` module.

`.VCD` Invokes the calculation of Vibrational Circular Dichroism (VCD) according to the implementation described in Ref. [78]. By default this is done using London orbitals in order to ensure fast basis set convergence as shown in Ref. [83]. The use of London orbitals can be disabled by the keyword `.NOLOND`.

Furthermore, the natural connection (Ref. [64, 82]) is default in order to ensure numerically stable results. The natural connection can be turned off by the keyword `.NODIFC` in which case the symmetric connection will be used.

In the current implementation, the keyword `.NOCMC` will be set true in calculations of Vibrational Circular Dichroism, that is, the coordinate system origin will be used as gauge origin. Changing this default value will give incorrect results for VCD.

Note that in the current release, VCD is not implemented for Density functional theory calculations, and the program will stop if VCD is requested for a DFT calculation.

`.VIBANA` Invokes a vibrational analysis in the current geometry. This will generate the vibrational frequencies in the current point. If combined with `.DIPGRA` the IR intensities will be calculated as well.

`.VROA` Invokes the calculation of Vibrational Raman Optical Activity, as described in Ref. [42]. This property needs a lot of settings in order to perform correctly, and the reader is therefore refered to Section 11.4, where the calculation of this property is described in more detail.

.WRTINT Forces the magnetic first-derivate two-electron integrals to be written to disc. This is default in MCSCF calculations, but not for SCF runs. This file can be very large, and it is not recommended to use this option for ordinary SCF runs.

25.1.2   Calculation of Atomic Axial Tensors (AATs): *AAT

Directives for controlling the calculation of Atomic Axial Tensors, needed when calculating Vibrational Circular Dichroism (VCD).

.INTPRI
>   READ (LUCMD,*) INTPRI
>
> Set the print level in the calculation of the necessary differentiated integrals when calculating Atomic Axial Tensors. Read one more line containing print level. Default value is value of `IPRDEF` from the general input module. The print level of the rest of the calculation of Atomic Axial Tensors are controled by the keyword `.PRINT` .

.NODBDR Skip contributions originating from first half-differentiated overlap integrals with respect to both nuclear distortions as well as magnetic field. This will give wrong results for VCD. Mainly for debugging purposes.

.NODDY Checks the calculation of the electronic part of the Atomic Axial Tensors by calculating these both in the ordinary fashion as well as by a noddy routine. The program will not perform a comparison, and will not abort if differences is found. Mainly for debugging purposes.

.NOELC Skip the calculation of the pure electronic contribution to the Atomic Axial Tensors. This will give wrong results for VCD. Mainly for debugging purposes.

.NONUC Skip the calculation of the pure nuclear contribution to the Atomic Axial Tensors. This will give wrong results for VCD. Mainly for debugging purposes.

.NOSEC Skip the calculation of second order orbital contributions to the Atomic Axial Tensors. This will give wrong results for VCD. Mainly for debugging purposes.

.PRINT
>   READ (LUCMD,*) IPRINT
>
> Set print level in the calculation of Atomic Axial Tensors (this does not include the print level in the integral calculation, which are controled by the keyword `.INTPRI`). Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

.SKIP Skips the calculation of Atomic Axial Tensors. This will give wrong results for VCD, but may be of interest for debugging purposes.

.STOP Stops the entire calculation after finishing the calculation of the Atomic Axial Tensors. Mainly for debugging purposes.

25.1.3 Linear response calculation: `*ABALNR`

Directives to control the calculation of frequency dependent linear response functions.

.FREQUE
```
READ (LUCMD,*) NFRVAL
READ (LUCMD,*) (FRVAL(I), I = 1, NFRVAL)
```

Set the number of frequencies as well as the frequency at which the frequency-dependent linear response equations are to be evaluated. Read one more line containing the number of frequencies to be calculated, and another line reading these frequencies. The frequencies are to be entered in atomic units. By default only the static case is evaluated.

.LIFETIME
```
READ (LUCMD,*) LIFETIME
```

Sets the lifetime of the excited states if absorption is also included in the calculation of the linear response functions as described in Ref. [12, 156]. The default is that no absorption is included in the calculation. The lifetime is given in cm^{-1}.

.MAX IT
```
READ (LUCMD,*) MAXITE
```

Set the maximum number of micro iterations in the iterative solution of the frequency-dependent linear response functions. Read one more line containing maximum number of micro iterations. Default value is 60.

.MAXPHP
```
READ (LUCMD,*) MXPHP
```

Set the maximum dimension for the sub-block of the configuration Hessian that will be explicitly inverted. Read one more line containing maximum dimension. Default value is 0.

.MAXRED
```
READ (LUCMD,*) MXRM
```

Set the maximum dimension of the reduced space to which new basis vectors are added as described in Ref. [22]. Read one more line containing maximum dimension. Default value is 400.

.OPTORB Use optimal orbital trial vectors in the iterative solution of the frequency-dependent linear response equations. These are generated as described in Ref. [22] by solving the orbital response equation exact, keeping the configuration part fixed.

.PRINT

      `READ (LUCMD,*) IPRLNR`

Set the print level in the calculation of frequency-dependent linear response properties. Read one more line containing the print level. The default value is the value of `IPRDEF` from the general input module.

.SKIP Skip the calculation of the frequency-dependent response functions. This will give wrong results for ROA. Mainly for debugging purposes.

.STOP Stops the program after finishing the calculation of the frequency-dependent linear response equations. Mainly for debugging purposes.

.THRESH

      `READ (LUCMD,*) THCLNR`

Set the convergence threshold for the solution of the frequency dependent response equations. Read one more line containing the convergence threshold (D12.6). The default value is $5.0 \cdot 10^{-5}$.

25.1.4 Dipole moment and dipole gradient contributions: *DIPCTL

Directives controlling the calculation of contributions to the dipole gradient appear in the `*DIPCTL` section.

.NODC Neglect contributions to traces from inactive one-electron density matrix. This will give wrong results for the dipole gradient. Mainly for debugging purposes.

.NODV Neglect contributions to traces from active one-electron density matrix. This will give wrong results for the dipole gradient. Mainly for debugging purposes.

.PRINT

      `READ (LUCMD,*) IPRINT`

Set print level in the calculation of the dipole gradient. Read one more line containing print level. The default is the variable `IPRDEF` from the general input module.

.SKIP Skip the calculation of dipole gradient.

.STOP Stop the program after finishing the calculation of the dipole gradient. Mainly for debugging purposes.

25.1.5 End of input: *END OF

The last directive in the input should be `*END OF`.

25.1.6 Calculation of excitation energies: *EXCITA

Directives to control the calculations of electronic transition properties and excitation energies appear in the `*EXCITA` input module. For SCF wave functions the properties are calculated using the random phase approximation (RPA) and for MCSCF wave functions the multiconfigurational (MC)-RPA is used. In the case of Kohn–Sham DFT, time-dependent linear response theory is used in the adiabatic approximation with the exact functional kernel. Implemented electronic transition properties are at the moment:

1. Oscillator Strength which determines visible and UV absorption.

2. Rotatory Strength which determines Electronic Circular Dichroism (ECD).

3. Excitation Energies. These are always calculated when invoking the `.EXCITA` keyword in the general input module.

.DIPSTR Calculates the dipole strength, that is, the dipole oscillator strength which determine the visible and UV absorption, using the dipole length form.

.FNAC Calculate first-order non-adiabatic coupling matrix elements. This is not yet fully implemented in the ABACUS program package and must be combined with subsequent RESPONSE-calculations.

.INTPRI
>   READ (LUCMD, *) IPRINT

> Set the print level in the calculation of the necessary differentiated integrals when calculating the linear response functions. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module. The print level of the rest of the calculation of electronic excitation energies are controled by the keyword `.PRINT` .

.MAX IT
>   READ (LUCMD,*) MAXITE

Set the maximum number of micro iterations in the iterative solution of the linear response equations. Read one more line containing maximum number of micro iterations. Default value is 60.

.MAXPHP

    READ (LUCMD,*) MXPHP

Set the maximum dimension for the sub-block of the configuration Hessian that will be explicitly inverted. Read one more line containing maximum dimension. Default value is 0.

.MAXRED

    READ (LUCMD,*) MXRM

Set the maximum dimension of the reduced space to which new basis vectors are added as described in Ref. [22]. Read one more line containing maximum dimension. Default value is 400.

.NEXCIT

    READ (LUCMD,*) (NEXCIT(I), I= 1,NSYM)

Set the number of excitation energies to be calculated in each symmetry. Read one more line containing the number of excitations in each of the irreducible representations of the molecular point group. The default is not to calculate any excitation energies in any of the irreducible representations.

.OPTORB Use optimal orbital trial vectors in the iterative solution of the eigenvalue equations. These are generate by solving the orbital response equation exact, keeping the configuration part fixed as described in Ref. [22].

.PRINT

    READ (LUCMD,*) IPREXE

Set the print level in the calculation of electronic excitation energies. Read one more line containing the print level. The default value is the `IPRDEF` from the general input module.

.ROTVEL Calculate rotational strengths in Electronic Circular Dichroism (ECD) without using London orbitals.

.SKIP Skip the calculation of electronic excitation energies. This will give wrong results for ECD. Mainly for debugging purposes.

.STOP Stops the program after finishing the calculation of the eigenvalue equations. Mainly for debugging purposes.

`.SUMRUL` Calculate oscillator strength sum rules.

`.THRESH`

>   READ (LUCMD,*) THREXC

> Set the convergence threshold for the solution of the linear response equations. Read one more line containing the convergence threshold. The default value is $1 \cdot 10^{-4}$.

`.TRIPLE` Indicates that it is triplet excitation energies that is to be calculated.

25.1.7   One-electron expectation values: `*EXPECT`

Directive that control the calculation of one-electron expectation values appear in the `*EXPECT` input module. Notice, however, that the directives controling the calculation of one-electron expectation values to the geometric Hessian appear in the `*ONEINT` section.

`.ALL CO` Indicates that all components of the expectation value contributions to the nuclear shielding or indirect spin–spin coupling tensors are to be calculated at the same time. This is the default for ordinary calculations. However, in direct and parallel calculations on large molecules this may give too large memory requirements, and instead only the components of one symmetry-independent nucleus are calculated at a time. However, by invoking this keyword, all components are calculated simultaneously even in direct/parallel calculations.

`.DIASUS` Invokes the calculation of the one-electron contribution to the magnetizability expectation value. By default this is done using London atomic orbitals. Default value is `TRUE` if magnetizability has been requested in the general input module, otherwise `FALSE`.

`.ELFGRA` Invokes the calculation of the electronic contribution to the nuclear quadrupole moment coupling tensor (that is, the electric field gradient). Default value is `TRUE` if nuclear quadrupole coupling constants have been requested in the general input module, otherwise `FALSE`.

`.NODC` Do not calculate contributions from the inactive one-electron density matrix. This will give wrong results for the one-electron expectation values. Mainly for debugging purposes.

`.NODV` Do not calculate contributions from the active one-electron density matrix. This will give wrong results for the one-electron expectation values. Mainly for debugging purposes.

.NEFIEL Invokes the evaluation of the electric field at the individual nuclei. Default value is `TRUE` if spin-rotation constants have been requested in the general input module, otherwise `FALSE`. In the current implementation, symmetry dependent nuclei cannot be used when calculating this property.

.POINTS

> `READ (LUCMD,*) NPOINT`

> Set the number of integration points to be used in the Gaussian quadrature when evaluating the diamagnetic spin-orbit integrals. Default value is 40.

.PRINT

> `READ (LUCMD,*) MPRINT`

> Set print level in the calculation of one-electron expectation values. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

.QUADRU Calculates the electronic contribution to the molecular (traceless) quadrupole moment. Default value is `TRUE` if molecular quadrupole moment have been requested in the general input module, otherwise `FALSE`.

.SHIELD Invokes the calculation of the one-electron contribution to the nuclear shielding expectation values. By default this is done using London atomic orbitals. Default value is `TRUE` if nuclear shieldings have been requested in the general input module, otherwise `FALSE`.

.SKIP Skip the calculation of one-electron expectation values. This may give wrong final results for some properties. Mainly for debugging purposes.

.SPIN-S Invokes the calculation of the diamagnetic spin-orbital integral, which is the diamagnetic contribution to indirect nuclear spin-spin coupling constants. Default value is `TRUE` if spin-spin couplings have been requested in the general input module, otherwise `FALSE`.

.STOP Stop the entire calculation after finishing the calculation of one-electron expectation values. Mainly for debugging purposes.

25.1.8 Geometry analysis: *GEOANA

Directives controlling the calculation and printing of bond angles and dihedral angles appear in the `*GEOANA` section. The program will also define atoms to be bonded to each other depending on their bond distance. For all atoms defined to be bonded to each other, the bond distance and bond angles will be printed.

```
.ANGLES
     READ (LUCMD,*) NANG
        DO 310 I = 1, NANG
           READ (LUCMD,*) (IANG(J,I), J = 1,3)
     310 CONTINUE
```

Calculate and print bond angles. Read one more line specifying the number of angles, and then read `NANG` lines containing triplets A, B, C of atom labels, each specifying a particular bond angle $\angle ABC$. Notice that in the current version of the program there is an upper limit of 20 bond angles that will be printed. The rest will be ignored. We also note that program always will print the angles between atoms defined to be bonded to each other on the basis of the van der Waals radii of the atoms.

```
.DIHEDR
     READ (LUCMD,*) NDIHED
        DO 410 I = 1, NDIHED
           READ (LUCMD,*) (IDIHED(J,I), J = 1,4)
     410 CONTINUE
```

Calculate and print dihedral (torsional) angles. Read one more line specifying the number of angles, and then read `NDIHED` lines containing quadruplets A, B, C, D of atom labels. The angle computed is that between the planes ABC and BCD. Notice that in the current version of the program there is an upper limit of 20 dihedral angles that will be printed. The rest will be ignored.

.SKIP Skip the geometry analysis, with the exceptions mentioned in the introduction to this section. This is the default value, but it is overwritten by the keywords `.ANGLES` and `.DIHEDR`.

25.1.9   Right-hand sides for response equations: *GETSGY

Directives affecting the construction of the right-hand sides (RHS)—that is, property gradient terms—for the response calculation as well as some matrices needed for reorthonormalization contributions appear in the `*GETSGY` section.

.ALLCOM Requests that all paramagnetic spin-orbit right-hand sides are to be calculated in one batch, and not for each symmetry-independent center at a time which is the default. This will slightly speed up the calculation, at the cost of significantly larger memory requirements.

```
.FCKPRI
     READ (LUCMD,*) IPRFCK
```

Set print level for the calculation of derivative Fock matrices. Read one more line specifying print level. The default is the value of `IPRDEF` in the general input module.

.FCKSKI Skip the derivative Fock matrix contributions to the right-hand sides. This will give wrong results for all properties depending on right hand sides. Mainly for debugging purposes.

.FCKTES Test the Fock matrices. Mainly for debugging purposes.

.FSTTES Test one-index transformation of derivative Fock matrices.

.GDHAM Write out differentiated Hamiltonian and differentiated Fock matrices to file for use in post-DALTON programs.

.GDYPRI

    READ (LUCMD,*) IPRGDY

Set print level for the calculation of the Y-matrix appearing in the reorthonormalization terms, as for instance in Ref. [22]. Default is the value of `IPRALL` defined by the `.PRINT` keyword. If this has not been specified, the default is the value of `IPRDEF` from the general input section.

.GDYSKI Skip the calculation of the lowest-order reorthonormalization contributions to the second-order molecular properties. This will give wrong results for these properties. Mainly for debugging purposes.

.INTPRI

    READ (LUCMD, *) IPRINT, IPRNTA, IPRNTB, IPRNTC, IPRNTD

Set print level for the derivative integral calculation for a particular shell quadruplet. Read one more line containing print level and the four shell indices. The print level is changed from the default for this quadruplet only. Default value is the value of `IPRDEF` from the general input module. Note that the print level of all shell quadruplets can be changed by the keyword `.PRINT` .

.INTSKI Skip the calculation of derivative integrals. This will give wrong results for the total molecular Hessian. Mainly for debugging purposes.

.NODC Do not calculate contributions from the inactive one-electron density matrix. This will give wrong results for the total molecular property. Mainly for debugging purposes.

.NODDY Test the orbital part of the right-hand side. The run will not be aborted. Mainly for debugging purposes.

.NODPTR The transformation of the two-electron density matrix is back-transformed to atomic orbital basis using a noddy-routine for comparison.

.NODV Do not calculate contributions from the active one-electron density matrix. This will give wrong results for the molecular property. Mainly for debugging purposes.

.NOFD Do not calculate the contribution from the differentiated Fock-matrices to the total right-hand side. This will give wrong results for the requested molecular property. Mainly for debugging purposes.

.NOFS Do not calculate the contribution to the total right-hand side from the one-index transformed Fock-matrices with the differentiated connection matrix. This will give wrong results for the requested molecular property. Mainly for debugging purposes.

.NOH1 Do not calculate the contribution from the one-electron terms to the total right-hand side. This will give wrong results for the requested property. Mainly for debugging purposes.

.NOH2 Do not calculate the contribution from the two-electron terms to the total right-hand side. This will give wrong results for the requested molecular property. Mainly for debugging purposes.

.NOORTH Do not calculate the orbital reorthonormalization contribution (the one-index transformed contributions) to the total right-hand side. This will give wrong results for the requested molecular property. Mainly for debugging purposes.

.NOPV Do not calculate contributions from the two-electron density matrix. This will give wrong results for the requested molecular property. Mainly for debugging purposes.

.NOSSF Do not calculate the contribution to the total right-hand side from the double-one-index transformation between the differentiated connection matrix and the Fock-matrix. This option will only affect the calculation of the molecular Hessian, and will give a wrong result for this. Mainly for debugging purposes.

.PRINT
>   READ (LUCMD,) IPRALL

>   Set print levels. Read one more line containing the print level for this part of the calculation. This will be the default print level in the calculation of differentiated two-electron integrals, differentiated Fock-matrices, derivative overlap matrices, two-electron density and derivative integral transformation, as well as in the construction of the right-hand sides. To set the print level in each of these parts individually, see the keywords `.FCKPRI`, `.GDYPRI`, `.INTPRI`, `.PTRPRI` and `.TRAPRI`.

.PTRPRI

    READ (LUCMD,) IPRTRA

Set print level for the two-electron densities transformation. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module. Note also that this print level is also controled by the keyword `.PRINT` .

.PTRSKI Skip transformation of active two-electron density matrix. This will give wrong results for the total second-order molecular property. Mainly for debugging purposes.

.RETURN Stop after the shell quadruplet specified under `.INTPRI` above. Mainly for debugging purposes.

.SDRPRI

    READ (LUCMD,) IPRSDR

Set the print level in the calculation of the differentiated connection matrix. Read one more line containing the print level. Default value is the value given by the keyword `.PRINT` . If this keyword has not been given, the default is the value of `IPRDEF` given in the general input module.

.SDRSKI Do not calculate the differentiated connection matrices. This will give wrong results for properties calculated with perturbation dependent basis sets. Mainly for debugging purposes.

.SDRTES The differentiated connection matrices will be transformed and printed in atomic orbital basis. Mainly for debugging purposes.

.SIRPR4

    READ (LUCMD, *) IPRI4

SIRIUS "output unit 4" print level. Read one more line specifying print level. Default is 0.

.SIRPR6

    READ (LUCMD, *) IPRI6

SIRIUS "output unit 6" print level. Read one more line specifying print level. Default is 0.

.SKIP Skip the calculation of right-hand sides. This will give wrong values for the requested second-order properties. Mainly for debugging purposes.

.SORPRI

    READ (LUCMD,*) IPRSOR

> Set print level for the two-electron density matrix sorting. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

.STOP Stop the the entire calculation after finishing the construction of the right-hand side. Mainly for debugging purposes.

.TIME Provide detailed timing breakdown for the two-electron integral calculation.

.TRAPRI

> `READ (LUCMD,*) IPRTRA`

> Set print level for the derivative integrals transformation. Read one more line specifying print level. Default is the value of `IPRDEF` from the general input module. Notice that the default print level is also affect by the keyword `.PRINT` .

.TRASKI Skip transformation of derivative integrals. Mainly for debugging purposes.

.TRATES Testing of derivative integral transformation. The calculation will not be aborted. Mainly for debugging purposes.

25.1.10 Linear response calculation: `*LINRES`

Directives to control the calculation of frequency-independent linear response functions. At present these directives only affect the calculation of frequency-independent linear response functions appearing in connection with singlet, magnetic imaginary perturbations.

.MAX IT

> `READ (LUCMD,*) MAXITE`

> Set the maximum number of micro iterations in the iterative solution of the frequency independent linear response functions. Read one more line containing maximum number of micro iterations. Default value is 60.

.MAXPHP

> `READ (LUCMD,*) MXPHP`

> Set the maximum dimension of the sub-block of the configuration Hessian that will be explicitly inverted. Read one more line containing maximum dimension. Default value is 0.

.MAXRED

> `READ (LUCMD,*) MXRM`

> Set the maximum dimension of the reduced space to which new basis vectors are added as described in Ref. [22]. Read one more line containing maximum dimension. Default value is 400.

.OPTORB Use optimal orbital trial vectors in the iterative solution of the frequency-independent linear response equations. These are generate by solving the orbital response equation exact, keeping the configuration part fixed as described in Ref. [22].

.PRINT

> READ (LUCMD,*) IPRCLC

> Set the print level in the solution of the magnetic frequency-independent linear response equations. Read one more line containing print level. Default is the value of IPRDEF in the general input module.

.SKIP Skip the calculation of the frequency-independent response functions. This will give wrong results for shielding, magnetizabilities, optical rotation, VCD, VROA and spin-spin coupling constants. Mainly for debugging purposes.

.STOP Stops the program after finishing the calculation of the frequency-independent linear response equations. Mainly for debugging purposes.

.THRESH

> READ (LUCMD,*) THRCLC

> Set the convergence threshold for the solution of the frequency-independent response equations. Read one more line containing the convergence threshold. The default value is $1.0 \cdot 10^{-4}$ for calculations which cannot take advantage of Sellers formula for quadratic errors in the response property [157], and $2.0 \cdot 10^{-3}$ for those calculations that can.

25.1.11 Nuclear contributions: *NUCREP

Directives affecting the nuclear contribution to the molecular gradient and molecular Hessian calculation appear in the *NUCREP section.

.PRINT

> READ (LUCMD,*) IPRINT

> Set the print level in the calculation of the nuclear contributions. Read one more line containing print level. Default value is the value of IPRDEF from the general input module.

.SKIP Skip the calculation of the nuclear contribution. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

.STOP Stop the program after finishing the calculation of the nuclear contributions. Mainly for debugging purposes.

25.1.12   One-electron integrals: `*ONEINT`

Directives affecting the calculation of one-electron integrals in the calculation of molecular gradients and molecular Hessians appear in the `*ONEINT` section.

`.NCLONE` Calculate only the classical contributions to the nuclear-attraction integrals.

`.NODC` Do not calculate contributions from the inactive one-electron density matrix. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

`.NODV` Do not calculate contributions from the active one-electron density matrix. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

`.PRINT`
> `READ (LUCMD,*) IPRINT`
>
> Set print level in the calculation of one-electron contributions to the molecular gradient and Hessian. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

`.SKIP` Skip the calculation of one-electron integral contributions to the molecular gradient and Hessian. This will give wrong total results for these properties. Mainly for debugging purposes.

`.STOP` Stop the entire calculation after the one-electron integral contributions to the molecular gradients and Hessians has been evaluated. Mainly for debugging purposes.

25.1.13   Relaxation contribution to Hessian: `*RELAX`

Directives controlling the calculation of the relaxation contributions (*i.e.* those from the response terms) to the different second-order molecular properties, appear in the `*RELAX` section.

`.NOSELL` Do not use Sellers' method [157]. This method ensures that the error in the relaxation Hessian is quadratic in the error of the response equation solution, rather than linear. Mainly for debugging purposes.

`.PRINT`
> `READ (LUCMD,*) IPRINT`
>
> Set the print level in the calculation of the relaxation contributions. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

.SKIP Skip the calculation of the relaxation contributions. This does not skip the solution of the response equations. This will give wrong results for a large number of second-order molecular properties. Mainly for debugging purposes.

.STOP Stop the entire calculation after the calculation of the relaxation contributions to the requested properties. Mainly for debugging purposes.

.SYMTES Calculate both the ij and ji elements of the relaxation Hessian to verify its Hermiticity or anti-Hermiticity (depending on the property being calculated). Mainly for debugging purposes.

25.1.14   Reorthonormalization contributions: *REORT

Directives affecting the calculation of reorthonormalization contributions to the geometric Hessian appear in the *REORT   section.

.PRINT
```
READ (LUCMD,*) IPRINT
```

Set print level in the calculation of the lowest-order reorthonormalization contributions to the molecular Hessian. Read one more line containing print level. Default value is the value of IPRDEF from the general input module.

.SKIP Skip the calculation of the reorthonormalization contributions to the molecular Hessian. This will give wrong results for this property. Mainly for debugging purposes.

.STOP Stop the entire calculation after finishing the calculation of the reorthonormalization contributions to the molecular Hessian. Mainly for debugging purposes.

25.1.15   Response calculation: *RESPON

Directives affecting the response (coupled-perturbed MCSCF) calculation of geometric perturbations appear in the *RESPON section.

.D1DIAG Neglect diagonal elements of the orbital Hessian when generating trial vectors. Mainly for debugging purposes.

.DONEXT Force the use of optimal orbital trial vectors in the solution of the geometric response equations as described in Ref. [22]. This is done by solving the orbital part exact while keeping the configuration part fixed.

.MAX IT
```
READ (LUCMD,*) MAXNR
```

Maximum number of iterations to be used when solving the geometric response equations. Read one more line specifying value. Default value is 60.

.MAXRED

    `READ (LUCMD,*) MAXRED`

Set the maximum dimension of the reduced space to which new basis vectors are added as described in Ref. [22]. Read one more line containing maximum dimension. Default value is the maximum of 400 and 25 times the number of symmetry-independent nuclei.

.MAXSIM

    `READ (LUCMD,*) MAXSIM`

Maximum number of geometric perturbations to solve simultaneously in a given symmetry. Read one more line specifying value. Default is 15.

.MCHESS Explicitly calculate electronic Hessian and test its symmetry. Does not abort the calculation. Mainly for debugging purposes.

.NEWRD Forces the solution vectors to be written to a new file. This will also imply that `.NOTRIA` will be set to `TRUE`, that is, that no previous solution vectors will be used as trial vectors.

.NOAVER Use an approximation to the orbital Hessian diagonal when generating trial vectors.

.NONEXT Do not use optimal orbital trial vectors.

.NOTRIA Do not use old solutions as trial vectors, even though they may exist.

.NRREST Restart geometric response calculation using old solution vectors.

.PRINT

    `READ (LUCMD,*) IPRINT`

Set the print level during the solution of the geometric response equations. Read one more line containing print level. Default value is the value of `IPRDEF` in the general input module.

.RDVECS

    `READ (LUCMD, *) NRDT`
    `READ (LUCMD, *) (NRDCO(I), I = 1, NRDT)`

Solve for specific geometric perturbations only. Read one more line specifying number to solve for and then another line specifying their sequence numbers. This may give

wrong results for some components of the molecular Hessian. Mainly for debugging purposes.

.SKIP Skip the solution of the geometric response equations. This will give wrong results for the geometric Hessian. Mainly for debugging purposes.

.THRESH

> READ (LUCMD,*) THRNR

> Threshold for convergence of the geometric response equations. Read one more line specifying value. Default value is 10^{-3}.

.STOP Stop the entire calculation after solving all the geometric response equations. Mainly for debugging purposes.

25.1.16 Second-order polarization propagation approximation: *SOPPA

This input module controls the calculation of molecular properties using the second-order polarization propagator approximation, see Chapter 18.

.HIRPA Use the higher-order RPA Polarization Propagator Approximation.

.SOPW4 Requests that the W4 term in the SOPPA expressions are calculated explicitly.

25.1.17 Indirect nuclear spin-spin couplings: *SPIN-S

This input module controls the calculation of which indirect nuclear spin-spin coupling constants and what contributions to the total spin-spin coupling constants that are to be calculated.

.ABUNDA

> READ (LUCMD,*) ABUND

> Set the natural abundance threshold in percent for discarding couplings between certain nuclei. By default all isotopes in the molecule with a natural abundance above this limit will be included in the list of nuclei for which spin-spin coupling constants will be calculated. Read one more line containing the abundance threshold in percent. The default value is 1.0 (*i.e.* 1%), which includes both protons and ^{13}C nuclei.

.COUPLING NUCLEUS

> READ (LUCMD,*) NUCSPI
> READ (LUCMD,*) (IPOINT(IS), IS=1,NUCSPI)

> Calculates all coupling constants in a molecule to a selected number of nuclei only. The first number NUCSPI is the number of nuclei to which couplings shall be calculated,

and the next line reads in the number of the symmetry-independent nucleus as given in the `MOLECULE.INP` file.

.ISOTOPE

    READ (LUCMD,*) (ISOTPS(IS), IS=1, NATOMS)

Calculate the indirect spin–spin coupling constants for a given isotopic constitution of the molecule. The next line reads the isotope number for each of the atoms in the molecule (including also symmetry-dependent molecules). The isotopic number for each atom is given in terms of the occurance in the list of natural abundance of the isotopes for the given atom, *i.e.* the most abundant isotope is number 1, the second-most abundant is number 2 and so on.

.NODSO Do not calculate diamagnetic spin-orbit contributions to the total indirect spin-spin coupling constants. This will give wrong results for the total spin-spin couplings.

.NOFC Do not calculate the Fermi contact contribution to the total indirect spin-spin coupling constants. This will give wrong results for the total spin-spin couplings.

.NOPSO Do not calculate the paramagnetic spin-orbit contribution to the indirect spin-spin coupling constants. This will give wrong results for the total spin-spin couplings.

.NOSD Do not calculate the spin-dipole contribution to the total indirect spin-spin coupling constants. This will give wrong results for the total spin-spin couplings.

.PRINT

    READ (LUCMD,*) ISPPRI

Set the print level in the output of the final results from the spin-spin coupling constants. In order to get all individual tensor components (in a.u.), a print level of at least 5 is needed. Read one more line containing the print level. Default value is the value of `IPRDEF` from the general input module.

.SD+FC Do not split the spin-dipole and Fermi contact contributions in the calculations.

.SDxFC ONLY Will only calculate the spin dipole–Fermi contact cross term, and the Fermi contact–Fermi contact contribution for the triplet responses. The first of these two terms only contribute to the anistotropy, and one may in this way obtain the most important triplet contributions to the isotropic and anisotropic spin-spin couplings by only solving one instead of seven response equations for each nucleus.

.SELECT

    READ (LUCMD,*) NPERT
    READ (LUCMD, *) (IPOINT(I), I = 1, NPERT)

Select which symmetry-independent nuclei for which indirect nuclear spin-spin couplings is to be calculated. This option will override any selection based on natural abundance (the `.ABUNDA` keyword), and at least one isotope of the nuclei requested will be evaluated (even though the most abundant isotope with a non-zero spin has a lower natural abundance than the abundance threshold). Read one more line containing the number of nuclei selected, and another line with their number (sorted after the input order). By default, all nuclei with an isotope with non-zero spin and with a natural abundance larger than the threshold will be included in the list of nuclei for which indirect spin-spin couplings will be calculated.

25.1.18   Translational and rotational invariance: `*TROINV`

Directives affecting the use of translational and rotational invariance [158] appear in the `*TROINV` section.

`.COMPAR` Use both translational and rotational symmetry and check the molecular Hessian against the Hessian obtained without the use of translational and rotational invariance. This is default in a calculation of vibrational circular dichroism (VCD).

`.PRINT`
   READ (LUCMD,*) IPRINT

   Set print level for the setting up and use of translational and rotational invariance. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module.

`.SKIP` Skip the setting up and use of translational and rotational invariance.

`.STOP` Stop the entire calculation after the setup of translational and rotational invariance. Mainly for debugging purposes.

`.THRESH`
   READ (LUCMD,*) THRESH

   Threshold defining linear dependence among supposedly independent coordinates. Read one more line specifying value. Default is 0.1.

25.1.19   Response equations for triplet operators: `*TRPRSP`

Directives controlling the set-up of right-hand sides for triplet perturbing operators (for instance the Fermi contact and spin-dipole operators entering the nuclear spin-spin coupling constants), as well as when solving the triplet response equations appear in the `*TRPRSP` input module.

`.INTPRI`

> `READ (LUCMD ,*) INTPRI`

> Set the print level in the calculation of the atomic integrals contributing to the different triplet operator right-hand sides. Read one more line containing the print level. Default is the value of `IPRDEF` from the general input module.

`.MAX IT`

> `READ (LUCMD,*) MAXTRP`

> Set the maximum number of micro iterations in the iterative solution of the triplet response equations. Read one more line containing the maximum number of iterations. Default is 60.

`.MAXPHP`

> `READ (LUCMD,*) MXPHP`

> Set the maximum dimension for the sub-block of the configuration Hessian that will be explicitly inverted. Read one more line containing maximum dimension. Default value is 0.

`.MAXRED`

> `READ (LUCMD,*) MXRM`

> Set the maximum dimension of the reduced space to which new basis vectors are added as described in Ref. [22]. Read one more line containing maximum dimension. Default value is 400.

`.NORHS` Skip the construction of the right-hand sides for triplet perturbations. As this by necessity implies that all right-hand sides and solution vectors are zero, this option is equivalent to `.SKIP` . This will furthermore give wrong results for the total spin-spin couplings. Mainly for debugging purposes.

`.NORSP` Skip the solution of the triplet response equations. This will give wrong results for the total spin-spin couplings. Mainly for debugging purposes.

`.OPTORB` Optimal orbital trial vectors used in the solution of the triplet response equations. These are generate by solving the orbital response equation exact, keeping the configuration part fixed as described in Ref. [22].

`.PRINT`

> `READ (LUCMD,*) IPRTRP`

> Set the print level during the setting up of triplet operator right-hand sides and in the solution of the response equations for the triplet perturbation operators. Read one

more line containing the print level. Default is the value of `IPRDEF` from the general input module.

.SKIP Skip the construction of triplet right-hand sides as well as the solution of the response equations for the triplet perturbation operators. This will give wrong results for the indirect nuclear spin-spin couplings. Mainly for debugging purposes.

.STOP Stop the entire calculation after generating the triplet right-hand sides, and solution of the triplet response equations. Mainly for debugging purposes.

.THRESH

    `READ (LUCMD,*) THRTRP`

Set the threshold for convergence in the solution of the triplet response equations. Read one more line containing the threshold. Default is $1 \cdot 10^{-4}$.

25.1.20 Two-electron contributions: *TWOEXP

Directives affecting the calculation of two-electron derivative integral contributions to the molecular gradient and Hessian appear in the `*TWOEXP` section.

.DIRTST Test the direct calculation of Fock matrices and integral distributions. Mainly for debugging purposes.

.FIRST Compute first derivative integrals but not second derivatives. This is default if only molecular gradients and not the molecular Hessian has been requested.

.INTPRI

    `READ (LUCMD,*) IPRINT, IPRNTA, IPRNTB, IPRNTC, IPRNTD`

Set print level for the derivative integral calculation for a particular shell quadruplet. Read one more line containing print level and the four shell indices. The print level is changed from the default for this quadruplet only. Default value is the value of `IPRDEF` from the general input module. Note that the print level of all shell quadruplets can be changed by the keyword `.PRINT` .

.INTSKI Skip the calculation of derivative integrals. This will give wrong results for the total molecular gradients and Hessians. Mainly for debugging purposes.

.NOCONT Do not contract derivative integrals (program back-transforms density matrices to the primitive Gaussian basis instead).

.NODC Do not calculate contributions from the inactive one-electron density matrix. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

.NODV Do not calculate contributions from the active one-electron density matrix. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

.NOPV Do not calculate contributions from the two-electron density matrix. This will give wrong results for the total molecular gradient and Hessian. Mainly for debugging purposes.

.PRINT

    `READ (LUCMD,*) IPRALL`

Set print levels. Read one more line containing the print level for this part of the calculation. This will be the default print level in the two-electron density matrix transformation, the symmetry-orbital two-electron density matrix sorting, as well as the print level in the integral derivative evaluation. To set the print level in each of these parts individually, see the keywords `.INTPRI`, `.PTRPRI`, `.SORPRI`.

.PTRNOD The transformation of the two-electron density matrix is back-transformed to the atomic orbital basis using a noddy-routine for comparison.

.PTRPRI

    `READ (LUCMD,*) IPRPRT`

Set print level for the two-electron density matrix transformation. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module. Note also that this print level is controled by the keyword `.PRINT` .

.PTRSKI Skip transformation of active two-electron density matrix. This will give wrong results for the total molecular Hessian. Mainly for debugging purposes.

.RETURN Stop after the shell quadruplet specified under `.INTPRI` above. Mainly for debugging purposes.

.SORPRI

    `READ (LUCMD,*) IPRSOR`

Set print level for the two-electron density matrix sorting. Read one more line containing print level. Default value is the value of `IPRDEF` from the general input module. Note also that this print level is controled by the keyword `.PRINT` .

.SORSKI Skip sorting of symmetry-orbital two-electron density matrix. This will give wrong results for the total molecular Hessian. Mainly for debugging purposes.

.SECOND Compute both first and second derivative integrals. This is default when calculating molecular Hessians.

.SKIP Skip all two-electron derivative integral and two-electron density matrix processing.

.STOP Stop the the entire calculation after finishing the calculation of the two-electron derivative integrals. Mainly for debugging purposes.

.TIME Provide detailed timing breakdown for the two-electron integral calculation.

25.1.21 Vibrational analysis: *VIBANA

Directives controlling the calculation of harmonic vibrational frequencies appear in the *VIBANA section, as well as properties depending on a normal coordinate analysis or vibrational frequencies. Such properties include in the present version of the program: Vibrational Circular Dichroism (VCD), Raman intensities, Raman Optical Activity (ROA), and vibrational averaging.

.HESFIL Read the molecular Hessian from the file DALTON.HES. This file may have been made in an earlier calculation using the keyword .HESPUN, or constructed from a calculation with the GaussianXX program and converted to DALTON format using the FChk2HES.f program. Useful in VCD and VROA analyses.

.HESPUN Write the molecular Hessian to the file DALTON.HES for use as a starting Hessian in first-order geometry optimizations (see keyword .HESFIL in the *OPTIMIZE input module), or for later use in a vibrational analysis (see keyword .HESFIL in this input module).

.ISOTOP READ (LUCMD,*) NISOTP, NATM
    DO 305 ICOUNT = 1, NISOTP
      READ (LUCMD,*) (ISOTP(ICOUNT,N), N = 1, NATM)
    END DO

Read in the number of different isotopically substituted species NISOTP for which we are to do a vibrational analysis. The isotopic species containing only the most abundant isotopes is always calculated.

NATM is the total number of atoms in the molecules (see discussion in Section 8.1). For each isotopic species, the isotope for each atom in the molecule is read in. A 1 denotes the most abundant isotope, a 2 the second-most abundant isotope and so on.

.PRINT
    READ (LUCMD,*) PRINT

Set the print level in the vibrational analysis of the molecule. Read one more line containing print level. Default value is the value of IPRDEF from the general input module.

.SKIP Skip the analysis of the vibrational frequencies and normal modes of the molecule.

Chapter 26

Linear and non-linear response functions, RESPONSE

26.1 Directives for evaluation of molecular response functions

The directives in the following subsections may be included in the input to RESPONSE. They are organized according to the program section names in which they appear.

RESPONSE is the most general part of the code for calculating many different electronic linear, quadratic, or cubic molecular response properties based on SCF, MCSCF or CI wave functions, as well as Kohn–Sham-based time-dependent density functional theory. No nuclear contributions are added.

If the final wave function from **WAVE FUNCTIONS was .CI, then a configuration interaction response calculation will be performed. This is equivalent to a CI sum-over-states calculation of response properties, but of course calculated directly without diagonalization of the full CI Hamiltonian matrix.

Some of the SCF/MCSCF response properties can also be requested from **PROPERTIES input modules. NOTE: for such properties you should request them either here or in **PROPERTIES, otherwise you will calculate them twice! Usually the output is nicest in the **PROPERTIES module (*e.g.* collected in tables and in commonly used units, most properties are only given in atomic units in RESPONSE), and nuclear contributions are included if relevant. Some specific properties, especially those involving nuclear derivatives, can only be calculated via **PROPERTIES.

Calculations of coupled cluster response properties are performed by different modules and are described in Chapter 27 on coupled cluster calculations.

In addition, SOPPA (Second-Order Polarization Propagator approximation) or SOPPA(CCSD) (Second Order Polarization Propagator Approximation with Coupled Cluster Singles and

Doubles Amplitudes) for the calculation of linear response properties and excitation energies with transition moments may be requested in this input section. The current implementation of the SOPPA method is described in Ref. [51] and of the SOPPA(CCSD) method in Ref. [53]. Note that a SOPPA calculation requires the keyword `.SOPPA`, whereas a SOPPA(CCSD) calculation requires the keyword `.SOPPA(CCSD)`.

26.1.1   General: `**RESPONSE`

General-purpose directives are given in the `**RESPONSE` section.

After the last directive of the `**RESPONSE` input group should follow another `**<something>` input group (or `**END OF DALTON INPUT` if this was the last input to DALTON).

`.HIRPA` Invoke the higher RPA approximation for the calculation of linear response properties. This approximation is identical to that of McKoy and coworkers [159, 160]. The requirements to the preceding wave function calculation is the same as for the `.SOPPA` keyword. This keyword overrides a simultaneous specification of `.SOPPA`.

`.INPTEST` Input test. For debugging purposes only. The program stops after the input section.

`.MAXPHP`
    `READ *, MAXPHP`
    Change the maximum dimension of H_0 subspace. Default is 100. PHP is a subblock of the CI matrix which is calculated explicitly in order to obtain improved CI trial vectors compared to the straight Davidson algorithm[161]. The configurations corresponding to the lowest diagonal elements are selected, unless `.PHPRESIDUAL` is specified. MAXPHP is the maximum dimension of PHP, the actual dimension will be less if MAXPHP will split degenerate configurations.

`.MAXRM`
    `READ *, MAXRM`
    Change the maximum dimension of the reduced space. Default is 200. When solving a linear system of equations or an eigenvalue equation, the reduced space is increased by the number of frequencies/excitations in each iteration. For single root calculations this should exceed the number of iterations required. MAXRM should be increased if many frequencies or excitation energies are to be calculated. Sharp convergence thresholds also require more iterations and thus larger dimension of the reduced space.

`.NOAVDI` Do not use Fock type decoupling of the two-electron density matrix. Add $F^I D$ instead of $(F^I + F^A)D$ to $E^{[2]}$ approximate orbital diagonal. Not recommended as the approximate orbital diagonal normally will become more different from the exact orbital diagonal.

.NODOIT Turns off direct one-index transformation [162]. In this way all one-index transformed integrals are stored on disk.

.NOITRA No integral transformation. Normally the two-electron integrals are transformed in the beginning of a response calculation. In some cases this is not desirable, *e.g.*, if the response part is only used for calculating average values of an operator, or if the transformed two-electron integrals already exist from a previous response calculation.

.OPTORB Orbital trial vectors are calculated with optimal orbital trial vector algorithm [22].

.ORBSFT
>   READ *, ORBSFT

Change the amount for shifting the orbital diagonal of the MCSCF Hessian. May be used if there is a large number of negative eigenvalues. Default is 10^{-4}.

.ORBSPC Calculation with only orbital operators.

.PHPRESIDUAL Select configurations for PHP matrix based on largest residual rather than lowest diagonal elements.

.SOPPA Requests the second order polarization propagator approximation in the linear response module. The SOPPA flag requires that the preceding SIRIUS calculation has generated the MP2 correlation coefficients and written them to disk (set **.RUN RESPONSE** in ****DALTON** input as well as **.HF** and **.MP2** in ****WAVE FUNCTIONS**). See example input in Chapter 5.

.PROPAV
>   READ '(A)', LABEL

Property average. The average value of an electronic one-electron operator is calculated. (Thus, no nuclear contributions are added.) The line following this option must contain the label of the operator given in the integral property file. (See section 22.)

.QRREST Restart quadratic response calculation. Requires that all needed linear response solutions and excitation vectors are available on the RSPVEC file.

.SOMIX Sum rule is calculated in mixed representation, that is, calculate $N_e = \langle 0 \mid [r, p] \mid 0 \rangle$ provided that dipole length and velocity integrals are available on the property integral file (calculated with ****HERMIT** options **.DIPLEN** and **.DIPVEL**). The calculated quantity gives a measure of the quality of the basis set.

.SOPPA(CCSD) Requests the second order polarization propagator approximation with coupled cluster singles and doubles amplitudes in the linear response module. The

SOPPA(CCSD) flag requires that the preceding coupled cluster calculation has generated the CCSD amplitudes and written them to disk (set `.RUN RESPONSE` in `**DALTON` input, `.HF` and `.CC` in `**WAVE FUNCTIONS` and `.SOPPA(CCSD)` in `*CC INPUT`). See example input in Chapter 5.

`.SOPW4` Calculate explicitly the W4 term described by Oddershede *et al.* [50]. This term is already included in the normal SOPPA or SOPPA(CCSD) result, and used mostly for comparing to older calculations. Note that this keyword requires that `.SOPPA` or `.SOPPA(CCSD)` is set.

`.TRPFLG` Triplet flag. This option is set whenever triplet (spin-dependent) operators must be used in a response calculation [163, 164]. This flag forces triplet linear response for `*LINEAR`, both for second order properties and electronic excitations (without and with `.SINGLE RESIDUE`).

26.1.2 Linear response calculation: `*LINEAR`

A linear response [163, 165] calculation is performed for a given choice of operators, $-\langle\langle A; B \rangle\rangle_\omega$. (Note that *minus* the linear response properties are written to output.)

In the same RESPONSE calculation these linear response properties can be calculated together with excitation energies and with long range dispersion coefficients, but not together with quadratic or cubic response.

`.ABOCHK` Sets up the orbital part of the $E^{[2]}$ and $S^{[2]}$ used in solving the linear response equation. Mainly for debugging purposes.

`.DIPLEN` Sets A and B to dipole operators.

`.DIPLNX/Y/Z` Sets A and B to the X, Y, or Z component of the dipole length operators, respectively.

`.DIPMAG` Sets A and B to angular momentum operators.

`.DIPMGX/Y/Z` Sets A and B to the X, Y, or Z component of the angular momentum operators.

`.DIPVEL` Sets A and B to the dipole velocity operators.

`.DIPVLX/Y/Z` Sets A and B to the X, Y, or Z component of the dipole velocity operator, respectively.

`.FERMI` Sets A and B to be all Fermi-contact operators found on the file `AOPROPER`, *i.e.* all Fermi-contact operators requested in the `**INTEGRALS` input module.

.FREQUE

    `READ *, NFREQ`

    `READ *, FREQ(1:NFREQ)`

    Response equations are evaluated at given frequencies. Two lines following this option must contain 1) The number of frequencies, 2) Frequencies in atomic units. Remember to increase `.MAXRM` if many frequencies are specified.

.ISTOCK Selects the starting row in setting up the orbital parts of $E^{[2]}$ and $S^{[2]}$ using the keyword `.ABOCHK`. Default is 1. Mainly for debugging purposes.

.MAX IT

    `READ (LUCMD,*) MAXITL`

    Maximum number of iterations for solving a linear response equation. Default is 60.

.MAXITO

    `READ (LUCMD,*) MAXITO`

    Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.MAXOCK Selects the last row in setting up the orbital parts of $E^{[2]}$ and $S^{[2]}$ using the keyword `.ABOCHK`. Default is 6. Mainly for debugging purposes.

.PRINT

    `READ *,IPRLR`

    Sets print level for linear response module. Default is 2.

.PROPRT

    `READ '(A)', LABEL`

    Sets A and B to a given operator with label; LABEL. (The calculation of the operator must be specified to the integral module, see section 22.) This keyword may be repeated for different properties.

.PV PSO Sets A and B in the linear response function to the parity-violating operator and the complete list of paramagnetic spin-orbit integrals. The `.TRPFLG` keyword will also be set by this option.

.PV SO Sets A and B in the linear response function to the parity-violating operator and the complete list of paramagnetic spin-orbit integrals. The `.TRPFLG` keyword will also be set by this option.

.PV SO1 Sets A and B in the linear response function to the parity-violating operator and the one-electron spin–orbit integrals. The `.TRPFLG` keyword will also be set by this option.

.PV SO2 Sets A and B in the linear response function to the parity-violating operator and the two-electron spin–orbit integrals. The .TRPFLG keyword will also be set by this option.

.QUADMOM Sets A and B to the quadrupole operators.

.QUADXX/XY/XZ/YY/YZ/ZZ Sets A and B to the XX, XY, XZ, YY, YZ, or ZZ component of the quadrupole operator, respectively.

.RESTLR Restart of response calculation. This can only be used if the operator specified is the same which was used *last* in the previous response calculation.

.SOPRSY Calculate both α_{ij} and α_{ji} to test the quadratic accuracy of the calculated property. Mainly for debugging purposes.

.FERMI Sets A and B to be all spin-dipole operators found on the file AOPROPER, *i.e.* all spin-dipole operators requested in the **INTEGRALS input module.

.SPIN-O Sets A and B to spin-orbit operators.

.SPNORX/Y/Z Sets A and B to the X, Y, or Z component of the spin–orbit operator, respectively.

.THCLR
    READ *, THCLR
    Relative convergence threshold for all requested linear response functions. Default is 1.0D-3; note that this number should be at least 10 times bigger than the final gradient norm in the SCF/MCSCF wave function optimization. The accuracy of the linear response properties will be quadratic in this threshold; thus the default corresponds to convergence to approximately 6 digits.

.TRIPLET Defines A and B to be triplet operators. Will also make a simultaneous *LINEAR .SINGLE RESIDUE calculation to a calculation of triplet excitation energies and transition moments.

26.1.3 Linear response excitation energies calculation: *LINEAR with .SINGLE RESIDUE

Single residues of the linear response function is computed. Residues of a linear response function correspond to transition moments and the associated poles correspond to vertical electronic excitation energies.

In the same RESPONSE calculation these excitation properties can be calculated together with linear response properties and with long range dispersion coefficients, but not together with quadratic or cubic response.

Required keywords:

.SINGLE RESIDUE Required to get excitation energies, without this keyword the linear response function will be evaluated, see Sec. 26.1.2.

Optional keywords

.ABCHK Sets up $E^{[2]}$ and $S^{[2]}$ used in solving the single residue linear response equation. Only for debugging purposes.

.ABSYM Tests the symmetry of $E^{[2]}$ and $S^{[2]}$ in the reduced space. Only for debugging purposes.

.ANTTES Test the antisymmetry of the single residue response vector. Only for debugging purposes.

.DIPLEN Sets A and B to dipole operators.

.DIPLNX/Y/Z Sets A and B to the X, Y, or Z component of the dipole length operators, respectively.

.DIPMAG Sets A and B to angular momentum operators.

.DIPMGX/Y/Z Sets A and B to the X, Y, or Z component of the angular momentum operators.

.DIPVEL Sets A and B to the dipole velocity operators.

.DIPVLX/Y/Z Sets A and B to the X, Y, or Z component of the dipole velocity operator, respectively.

.MAX IT
    READ *, MAXITP
    Maximum number of iterations for solving the single residue linear response eigenvalue equation. Default is 60.

.MAXITO
    READ *, MAXITO
    Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.NSTART

> `READ (LUCMD,*) (NPPSTV(J),J=1,NSYM)`
>
> The number of start vectors to be used in the optimization of the transition vectors in each symmetry. By default this is set equal to the number of excited states that have been requested through the keyword **.ROOTS**. It can be relevant to make the number of start vectors bigger, for example if the molecule has higher symmetry than used in the calculation. In this case one might need more start vectors to get a representative of each symmetry.

.NSIMUL

> `READ (LUCMD,*) (NPPSIM(J),J=1,NSYM)`
>
> The number of eigenvectors to solve simultaneously in each symmetry. Normally decided automatically by the program depending on available memory and size of eigenvectors.

.PRINT

> `READ *,IPRPP`
>
> Sets print level for single residue linear response module. Default is 2.

.PROPRT

> `READ '(A)', LABEL`
>
> Calculate either singlet or triplet transition moments for a given operator with label; LABEL. (The calculation of the operator must be specified to the integral module, see section 22.) This keyword may be repeated for different properties.

.QUADMOM Sets A and B to the quadrupole operators.

.QUADXX/XY/XZ/YY/YZ/ZZ Sets A and B to the XX, XY, XZ, YY, YZ, or ZZ component of the quadrupole operator, respectively.

.RESTPP Restart of single residue response calculation. This can only be used if the root which is specified is the same which was used *last* in the previous single residue response calculation.

.ROOTS

> `READ '*',(ROOTS(I) I=1,NSYM)`
>
> Number of roots. The line following this option contains the number of excited states per symmetry. Excitation energies are calculated for each state and if any operators are given, symmetry-allowed transition moments are calculated between the reference state and the excited states. Remember to increase **.MAXRM** if many roots are specified. Default: one of each symmetry.

.SPIN-O Sets A and B to spin-orbit operators. Warning: this option implies `.TRIPLET` and forces the excitations to be of triplet symmetry, and all operators—including *e.g.* `.DIPLEN`—will be assumed by the program to be of triplet symmetry!!

.SPNORX/Y/Z Sets A and B to the X, Y, or Z component of the spin–orbit operator, respectively. Warning: this option implies `.TRIPLET` and forces the excitations to be of triplet symmetry, and all operators—including *e.g.* `.DIPLEN`—will be assumed by the program to be of triplet symmetry!!

.THCPP
  `READ *, THCPP`
  Threshold for solving the single residue linear response eigenvalue equation. Default is 1.0D-3; note that this number should be at least 10 times bigger than the final gradient norm in the SCF/MCSCF wave function optimization, otherwise you may encounter numerical problems. The accuracy of the pole (excitation energy) will be quadratic in this threshold, thus the default corresponds to approximately 6 digits. The accuracy of transition moments will be linear in this threshold.

.TRIPLET Calculate triplet excitation energies and transition moments. Will also make a simultaneous linear response calculation of triplet symmetry.

.OLSEN CI trial vectors are obtained with Olsen algorithm.

26.1.4 Quadratic response calculation: `*QUADRA`

Calculation of third order properties as quadratic response functions. A, B, and C-named options refer to the operators in the quadratic response function $\langle\langle A; B, C \rangle\rangle_{\omega_b, \omega_c}$ [164, 166, 167]

  The second order properties from the linear response functions $\langle\langle A; B, \rangle\rangle_{\omega_b}$ are also printed (if A and B operators have the same spin symmetry), as they can be obtained at no extra computational cost.

.A2TEST Test the contributions to the quadratic response function arising from the $A^{[2]}$ term. Mainly for debugging purposes.

.APROP, .BPROP, .CPROP
  `READ(LUCMD,'( A )')LABEL`
  Specify the operators A, B, and C. The line following this option should be the label of the operator as it appears in the file AOPROPER.

  Note that giving the label `ANGMOM`, `1SPNORB`, `2SPNORB`, or `MNFSPNOR`, all the components of angular momentum, one-electron spin–orbit, two-electron spin–orbit or the atomic mean-field spin–orbit operator will be selected.

By specifying the labels `FERMI CO`, `SPIN-DIP` or `PSO`, all components of the Fermi contact, spin–dipole or paramagnetic spin–orbit integrals that can be found on the file `AOPROPER` will be selected. These integrals are selected by the appropriate keywords in the `**INTEGRALS` input module.

`.ASPIN, .BSPIN, .CSPIN`
    `READ(LUCMD,*)ISPINA`
    Spin information for quadratic response calculations. The line following these options contains the spin rank of the operators A, B, and C, respectively, 0 for singlet operators and 1 for triplet operators. In a triplet response calculations two of these operators are of rank one, and the remaining operator of rank zero.

`.BFREQ, .CFREQ`
    `READ (LUCMD,*) NBQRFR`
    `READ (LUCMD,*) (BQRFR(J),J=1,NBQRFR)`
    Individual specification of the frequencies ω_b and ω_c. Input as in `.FREQUE` above. May not be used for `.SHG` and `.POCKEL`. May not be used together with `.FREQUE`. Default is one frequency of each type: zero (static).

`.DIPLEN` Sets A, B, and C to dipole operators.

`.DIPLNX/Y/Z` Sets A, B, and C operators to the X, Y, or Z component of the dipole length operators, respectively.

`.E3TEST` Test the contributions to the quadratic response function arising from the $E^{[3]}$ and $S^{[3]}$ terms. Mainly for debugging purposes.

`.FREQUE`
    `READ *, NFREQ`
    `READ *, FREQ(1:NFREQ)`
    Response equations are evaluated at given frequencies. Two lines following this option must contain 1) The number of frequencies, 2) Frequencies. For the Kerr effect only the B-frequency is set, and in other cases both B and C-frequencies are set. May not be used together with `.BFREQ` or `.CFREQ`. Default is one frequency of each type: zero (static).

`.ISPABC`
    `READ *, ISPINA,ISPINB,ISPINC`
    Spin symmetry of A-operators (ISPINA), B-operators (ISPINB), and the excitation operator (ISPINC): "0" for singlet and "1" for triplet. Default is "0,0,0", *i.e.* all of singlet spin symmetry. **Note: triplet operators are only implemented for singlet reference states.**

.MAX IT Maximum number of iterations for solving a linear response equation. Default is 60.

.MAXITO Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.OPTREF Only response functions connected with optical rectification $\beta(0; \omega, -\omega)$, are computed. Can be specified together with **.SHG** and **.POCKEL**. Frequencies must be specified with **.FREQUE**. Remember to specify operators as well, *e.g.* **.DIPLEN**.

.POCKEL Only response functions connected with electro-optical Pockels effect $\beta(-\omega; \omega, 0)$, are computed. Can be specified together with **.SHG** and **.OPTREF**. Frequencies must be specified with **.FREQUE**. Remember to specify operators as well, *e.g.* **.DIPLEN**.

.PRINT
  READ *,IPRHYP
  Print level. Default is 2.

.REFCHK Check reference state(?).

.SHG Only response functions connected with second harmonic generation are computed, $\beta(-2\omega, \omega, \omega)$. Can be specified together with **.POCKEL**. Frequencies must be specified with **.FREQUE**. Remember to specify operators as well, *e.g.* **.DIPLEN**.

.SOSHIE Analyze the calculated response equations to give the quadratic response spin-orbit contributions to the nuclear shielding constants. Will report the spin-orbit corrections to the shieldings in ppm. Note that this keyword will not set up the required quadratic response functions, only analyze the calculated results if appropriate quadratic response functions have been requested.

.SOSPIN Analyze the calculated response equations to give the quadratic response spin-orbit contributions to the indirect spin–spin coupling constants. Will calculate the spin-orbit corrections to the reduced spin–spin coupling constants. Note that this keyword will not set up the required quadratic response functions, only analyze the calculated results if appropriate quadratic response functions have been requested.

.THCLR Threshold for solving the linear response equations. Default is 10^{-3}. The error in the calculated property is linear in this threshold.

.TSTJEP
  READ(LUCMD,*) IAABB
  Include only $\alpha - \alpha$ (IAABB=1) or $\alpha - \beta$ (IAABB=2) components of the active density in the construction of the quadratic response function. Mainly for debugging purposes.

.X2TEST Test the contributions to the quadratic response function arising from the $X^{[2]}$ term. Mainly for debugging purposes.

26.1.5 Quadratic response calculation of second order transition moments: *QUADRA with .SINGLE RESIDUE

.A2TEST Test the contributions to the quadratic response function arising from the $A^{[2]}$ term. Mainly for debugging purposes.

.APROP, .BPROP Specify the operators A and B, respectively. The line following this option should be the label of the operator as it appears in the file AOPROPER.

.BFREQ, .FREQUE
 READ *, NFREQ
 READ *, FREQ(1:NFREQ)
 The frequencies ω_b in atomic units. Response equations are evaluated at given frequencies. Two lines following this option must contain 1) The number of frequencies, 2) Frequencies.

.DIPLEN Sets A and B to x, y, z dipole operators.

.DIPLNX Sets A and B to the x dipole operator.

.DIPLNY Sets A and B to the y dipole operator.

.DIPLNZ Sets A and B to the z dipole operator.

.E3TEST Test the contributions to the quadratic response function arising from the $E^{[3]}$ and $S^{[3]}$ terms. Mainly for debugging purposes.

.ISPABC
 READ *, ISPINA,ISPINB,ISPINC
 Spin symmetry of A-operators (ISPINA), B-operators (ISPINB), and the excitation operator (ISPINC): "0" for singlet and "1" for triplet. Default is "0,0,0", *i.e.* all of singlet spin symmetry. **Note: triplet operators are only implemented for singlet reference states.**

.MAXITL Maximum number of iterations for linear equations in this section. Default is 60.

.MAXITP Maximum number of iterations in solving the linear response eigenvalue equations. Default is 60.

.MAXITO Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.MCDBTERM Specifies the calculation of all individual components to the $\mathcal{B}(0 \to f)$ term of magnetic circular dichroism (MCD). This keyword sets up the calculation so that no further response input is required except **.ROOTS**. The A operator is set equal to the α component of dipole operator and the B operator to the β component of the angular momentum operator. The resulting "mixed" two-photon transition moment to state f is then multiplied the dipole-allowed one-photon transition moment from state f (for the γ component, with $\alpha \neq \beta \neq \gamma$). [17]

.MNFPHO Specifies a phosphorescence calculation using the atomic mean-field approximation for the spin–orbit operator, *i.e.* the spin-orbit induced singlet-triplet transition. This keyword sets up the calculation so that no further response input is required except **.ROOTS**; the A operator is set to the dipole operators and the B operator is set to the atomic mean-field spin-orbit operators. The reference state *must* be a singlet spin state.

.PHOSPHORESCENCE Specifies a phosphorescence calculation, *i.e.* the spin-orbit induced singlet-triplet transition. This keyword sets up the calculation so that no further response input is required except **.ROOTS**; the A operator is set to the dipole operators and the B operator is set to the spin-orbit operators. [164, 168] The reference state *must* be a singlet spin state.

.PRINT
```
READ *,IPRSMO
```
Print level. Default is 2.

.ROOTS
```
READ '*',(ROOTS(I) I=1,NSYM)
```
Number of roots. The line following this option contains the number of excited states per symmetry. Excitation energies are calculated for each state and if any operators are given, symmetry-allowed second order transition moments are calculated between the reference state and the excited states. Remember to increase **.MAXRM** if many frequencies are specified.

.SINGLE RESIDUE Required to compute the single residue of the quadratic response function. For the case of dipole operators this corresponds to two-photon transition moments.

.THCLR
```
READ *, THCLR
```
Threshold for solving the linear response equations. Default is 10^{-3}.

`.THCPP`

    `READ *, THCPP`

    Threshold for solving the linear response eigenvalue equation. Default is 10^{-3}.

`.TWO-PHOTON` Sets up the calculation of the two-photon transition strengths. This calculates two-photon transition strengths for all the excited states requested by the keyword `.ROOTS`, calculating the necessary quadratic response functions using the half-frequency of the excitation energy to the given state.

`.X2TEST` Test the contributions to the quadratic response function arising from the $X^{[2]}$ term. Mainly for debugging purposes.

26.1.6 Quadratic response calculation of transition moments between excited states: `*QUADRA` with `.DOUBLE RESIDUE`

Calculation of third-order properties as quadratic response functions. A, B, and C-named options refer to the operators in the quadratic response function $\langle\langle A; B, C\rangle\rangle_{\omega_b,\omega_c}$ [164, 166, 167]

`.A2TEST` Test the contributions to the quadratic response function arising from the $A^{[2]}$ term. Mainly for debugging purposes.

`.DIPLEN` Sets A to dipole operators.

`.DIPLNX` Sets A to the x dipole operator.

`.DIPLNY` Sets A to the y dipole operator.

`.DIPLNZ` Sets A to the z dipole operator.

`.DIPMAG` Sets A and B to angular momentum operators.

`.DIPMGX/Y/Z` Sets A and B to the X, Y, or Z component of the angular momentum operators.

`.DOUBLE RESIDUE` Computes the double residue of the quadratic response function. Double residues of the quadratic response function correspond to transition moments between excited states, $\langle B \mid A \mid C \rangle$.

`.DIPVEL` Sets A and B to the dipole velocity operators.

`.DIPVLX/Y/Z` Sets A and B to the X, Y, or Z component of the dipole velocity operator, respectively.

.E3TEST Test the contributions to the quadratic response function arising from the $E^{[3]}$ and $S^{[3]}$ terms. Mainly for debugging purposes.

.EXMTES Test that the transition moment is symmetric, *i.e.* that $\langle i\,|A|\,j\rangle = \langle j\,|A|\,i\rangle$. Mainly for debugging purposes.

.IPREXM

```
READ *,IPREXM
```

Print level for special excited state transition moment routines.

.ISPABC

```
READ *, ISPINA,ISPINB,ISPINC
```

Spin symmetry of *A*-operators (ISPINA) and the left and right excitation operators (ISPINB and ISPINC): "0" for singlet and "1" for triplet. Default is "0,0,0", *i.e.* all of singlet spin symmetry. **Note: triplet operators are only implemented for singlet reference states.**

.MAX IT Maximum number of iterations for solving linear response eigenvalue equation in this section.

.MAXITO Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.PRINT

```
READ *,IPRPP
```

Print level for solving linear response eigenvalue equations.

.PROPRT Specify another *A* operator. The line following this option should be the label of the operator as it appears in the file AOPROPER. This option may be repeated for different property operators.

.QUADMOM Sets *A* and *B* to the quadrupole operators.

.QUADXX/XY/XZ/YY/YZ/ZZ Sets *A* and *B* to the XX, XY, XZ, YY, YZ, or ZZ component of the quadrupole operator, respectively.

.SPIN-O Sets *A* and *B* to spin-orbit operators. Warning: this option implies **.TRIPLET** and forces the excitations to be of triplet symmetry, and all operators—including *e.g.* **.DIPLEN**—will be assumed by the program to be of triplet symmetry!!

.SPNORX/Y/Z Sets *A* and *B* to the X, Y, or Z component of the spin–orbit operator, respectively. Warning: this option implies **.TRIPLET** and forces the excitations to be of triplet symmetry, and all operators—including *e.g.* **.DIPLEN**—will be assumed by the program to be of triplet symmetry!!

.THCPP

> READ *, THCPP
>
> Threshold for solving the linear response eigenvalue equation. Default is 10^{-3}.

.X2TEST Test the contributions to the quadratic response function arising from the $X^{[2]}$ term. Mainly for debugging purposes.

26.1.7 Cubic response calculation: *CUBIC

Calculation of fourth-order properties as cubic response functions [169, 170, 90]. A, B, C, and D-named options refer to the operators in the cubic response function $\langle\langle A; B, C, D \rangle\rangle_{\omega_b, \omega_c, \omega_d}$

.APROP, .BPROP, .CPROP, .DPROP Specify the operators A, B, C, and D. The line following this option should be the label of the operator as it appears in the file AOPROPER.

.BFREQ, .CFREQ, .DFREQ The frequencies ω_b, ω_c, and ω_d, respectively. Input as in .FREQUE.

.DC-SHG Only response functions connected to the static electric field-induced second harmonic generation are computed, $\gamma(-2\omega; \omega, \omega, 0)$.

.DC-KERR Only response functions connected to the static electric field induced Kerr effect are computed, $\gamma(-\omega; \omega, 0, 0)$.

.DIPLEN Sets A, B, C, and D to dipole operators.

.DIPLNX Sets A, B, C, and D to the x dipole operator.

.DIPLNY Sets A, B, C, and D to the y dipole operator.

.DIPLNZ Sets A, B, C and D to the z dipole operator.

.FREQUE

> READ *, NFREQ
>
> READ *, FREQ(1:NFREQ)
>
> Sets the frequencies whenever a optical process is specified. Can also be used for the residue calculation and in which case both ω_b and ω_c for the single residue and only ω_b for the double residue.

.IDRI Only response functions connected to the intensity dependent refractive index are computed, $\gamma(-\omega; \omega, -\omega, \omega)$.

.INVEXP Solve the linear set of equations for the second-order perturbed wave function through explicit matrix inversion. Mainly for debugging purposes.

`.ISPABC`
> `READ *, ISPINA,ISPINB,ISPINC`
> Spin symmetry of A, B, C, and D-operators (ISPINA/B/C/D), "0" for singlet and "1" for triplet. Note that currently only singlet triplet response functions have been implemented. Do not use.

`.MAX IT` Maximum number of iterations for solving linear equations, default value is 60.

`.MAXITO` Maximum number of optimal orbital trial vector microiterations, default value is 5.

`.PRINT` Print flag for output, default value is 2. Timer information is printed out if print flag greater than 5. Response vectors printed out if print flag greater than 10.

`.THCLR` Threshold for convergence of response vectors, default value is 10^{-3}.

`.THG` Only response functions connected to the third harmonic generation are computed, $\gamma(-3\omega; \omega, \omega, \omega)$ [171].

`.THRNRM` Threshold for norm of property vector $X^{[1]}$ to be considered to be greater than zero in order to solve the linear equation
$\left(E^{[2]} - S^{[2]}\right) N^X = X^{[1]}$, default value is 10^{-9}.

26.1.8 Cubic response calculation of third-order transition moments: *CUBIC with .SINGLE RESIDUE

Calculation of single residues of cubic response functions [169, 170, 90]. A, B, C, and D-named options refer to the operators in the cubic response function $\langle\langle A; B, C, D \rangle\rangle_{\omega_b, \omega_c, \omega_d}$

`.APROP, .BPROP, .CPROP` Specify the operators A, B, and C, respectively. The line following this option should be the label of the operator as it appears in the file AOPROPER.

`.BFREQ, .CFREQ` The frequencies ω_b and ω_c, respectively. Input as in `.FREQUE`.

`.DIPLEN` Sets A, B, C, and D to dipole operators.

`.DIPLNX` Sets A, B, C, and D to the x dipole operator.

`.DIPLNY` Sets A, B, C, and D to the y dipole operator.

`.DIPLNZ` Sets A, B, C and D to the z dipole operator.

.FREQUE
```
READ *, NFREQ
READ *, FREQ(1:NFREQ)
```
Sets the frequencies whenever a optical process is specified. Can also be used for the residue calculation in which case it sets both ω_b and ω_c for the single residue and only ω_b for the double residue.

.MAX IT Maximum number of iterations for solving linear equations, default value is 60.

.MAXITO Maximum number of optimal orbital trial vector microiterations, default value is 5.

.MAXITP Maximum number of iteration for solving eigenvalue equation, default value is 60.

.NOHG Do not restrict the calculation to the 'harmonic generation case', that is, allow a different number and different numerical values for the frequencies of the B and C operators. By default, it is assumed that the B and C operator frequencies are identical.

.PRINT Print flag for output, default value is 2. Timer information is printed out if print flag greater than 5. Response vectors printed out if print flag greater than 10.

.ROOTS Number of roots (excited states) to converge.
```
READ (LUCMD,*) (NTMCNV(J),J=1,NSYM)
```

.SINGLE Computes the single residue of the cubic response function. In the case of dipole operators this corresponds to three-photon absorption.

.THCLR Threshold for convergence of response vectors, default value is 10^{-3}.

.THCPP Threshold for convergence of eigenvector, default value is 10^{-3}.

.THREE-PHOTON Sets up the calculation of the three-photon transition strengths. This calculates two-photon transition strengths for all the excited states requested by the keyword **.ROOTS**, calculating the necessary cubic response functions using a third of the frequency of the excitation energy to the given state.

26.1.9 Cubic response calculation of second order moments between excited states and excited state polarizabilities: *CUBIC with .DOUBLE RESIDUE

Calculation of double residues of cubic response functions [169, 170, 90]. A, B, C, and D-named options refer to the operators in the cubic response function $\langle\langle A; B, C, D\rangle\rangle_{\omega_b,\omega_c,\omega_d}$. C and D refer to the left hand state and right hand state after the double residue has been taken.

Excited state polarizabilites are only calculated if one or more of the keywords .DIPLEN, .DIPLNX, .DIPLNY, and .DIPLNZ are specified. Only singlet excitations and singlet property operators are implemented.

.APROP, .BPROP Specify the operators A and B, respectively. The line following this option should be the label of the operator as it appears in the file AOPROPER. These two keywords can be repeated for different properties.

.BFREQ The frequencies ω_b. Input as in .FREQUE. Default only zero frequency (static).

.DIPLEN Sets A and B to dipole operators.

.DIPLNX Sets A and B to the x dipole operator.

.DIPLNY Sets A and B to the y dipole operator.

.DIPLNZ Sets A and B to the z dipole operator.

.DOUBLE REQUIRED. Computes the double residue of the cubic response function. In the case of dipole operators this corresponds to excited state polarizabilities and two-photon transition moments between excited states [171].

.FREQUE
    READ *, NFREQ
    READ *, FREQ(1:NFREQ)
    Sets the frequencies whenever a optical process is specified. Can also be used for the residue calculation and it does then set both ω_b and ω_c for the single residue and only ω_b for the double residue. Default only zero frequency (static).

.MAX IT Maximum number of iterations for solving linear equations, default value is 60.

.MAXITO Maximum number of optimal orbital trial vector microiterations, default value is 5.

.MAXITP Maximum number of iteration for solving eigenvalue equation, default value is 20.

.PRINT Print flag for output, default value is 2. Timer information is printed out if print flag greater than 5. Response vectors printed out if print flag greater than 10.

.ROOTS Number of roots (excited states) to converge for each spatial symmetry.
    READ (LUCMD,*) (NTPCNV(J),J=1,NSYM)
    Default: one of each symmetry.

.THCLR Threshold for convergence of response vectors, default value is 10^{-3}.

.THCPP Threshold for convergence of eigenvector, default value is 10^{-3}.

26.1.10   Module for C6, C8, C10 coefficients and more*C6

.C6ATM, .C8ATM, .C10ATM .C6ATM, .C8ATM, .C10ATM do the same as .C6SPH etc. for atoms. Only $M_L = 0$ is calculated and written to file (all M_L values give same multipole moment for atoms).

.C6LMO, .C8LMO, .C10LMO .C6LMO, .C8LMO, .C10LMO is .C6SPH etc. for linear molecules. Only multipole moments with zero or positive M_L are calculated and written to file.

.C6SPH, .C8SPH, .C10SPH Specification of one of .C6SPH, .C8SPH, .C10SPH calculates and writes to a formatted interface file (RESPONSE.C8) the spherical multipole moments in the specified/default grid points needed for C6, C8, and C10 coefficients, respectively ($L = 1$, $L = 1, 2, 3$, or $L = 1, 2, 3, 4, 5$; all for $M_L = -L, \ldots, 0, \ldots, L$).

.DIPLEN Sets A and B to dipole operators.

.DIPLNX/Y/Z Sets A and B to the X, Y, or Z component of the dipole length operators, respectively.

.DIPMAG Sets A and B to angular momentum operators.

.DIPMGX/Y/Z Sets A and B to the X, Y, or Z component of the angular momentum operators.

.DIPVEL Sets A and B to the dipole velocity operators.

.DIPVLX/Y/Z Sets A and B to the X, Y, or Z component of the dipole velocity operator, respectively.

.FREQUE
    READ *, NCFREQ
    READ *, CFREQ(1:NCFREQ)
    Response equations are evaluated at given frequencies. Two lines following this option must contain 1) The number of frequencies, 2) Frequencies in atomic units.

.GSLEGN Use a Gauss–Legendre grid for calculating imaginary polarizabilities.

.MAX IT

    `READ (LUCMD,*) MAXITC`

    Maximum number of iterations for solving a linear response equation. Default is 60.

.MAXITO

    `READ (LUCMD,*) MAXITO`

    Maximum number of iterations in the optimal orbital algorithm [22]. Default is 5.

.MAXMOM

    `READ (LUCMD,*) MAXMOM`

    The maximum order of the Cauchy moments calculated. The default order is 6.

.GRID

    `READ (LUCMD,*) NGRID`

    Read in the number of grid points to use in the numerical integration of the Cauchy moments. Default is 10.

.QUADMOM Sets A and B to the quadrupole operators.

.QUADXX/XY/XZ/YY/YZ/ZZ Sets A and B to the XX, XY, XZ, YY, YZ, or ZZ component of the quadrupole operator, respectively.

.PRINT

    `READ (LUCMD,*),IPRC6`

    The line following gives the print level for the calculation of Cauchy moments.

.PROPRT

    `READ '(A)', LABEL`

    Sets A and B to a given operator with label; LABEL. (The calculation of the operator must be specified to the integral module, see section 22.) This keyword may be repeated for different properties.

.THCC6

    `READ *, THCC6`

    Relative convergence threshold for all requested linear response functions. Default is 1.0D-3; note that this number should be at least 10 times bigger than the final gradient norm in the SCF/MCSCF wave function optimization.¡

Comments:

    You must tell the integral module to calculate the necessary one-electron integrals. For **.C8SPH**, **.C8ATM**, or **.C8LMO** you will need

```
**INTEGRALS
.SPHMOM
   3
```

which calculate spherical moments for $L = 0, \ldots, 3$. For the `.C6xxx` and the `.C10xxx` options you will need $L = 0, 1$ and $L = 0, \ldots, 5$, respectively.

26.1.11   Electron Spin Resonance: `*ESR`

Calculation of ESR parameters

Hyperfine coupling

Default: hyperfine coupling tensors using the Restricted-Unrestricted Approach.

`.FCCALC`

Calculate the isotropic Fermi-contact contributions to hyperfine coupling tensors

`.SDCALC`

Calculate the spin-dipole contributions to the hyperfine coupling tensor

`.ATOMS`

```
READ (LUCMD,*),ESRNUC
READ (LUCMD,*) (NUCINF(IG), IG = 1, ESRNUC)
```

Select atoms for which to calculate hyperfine coupling constants. The first line contains the number of atoms and the second line the index of each atom (ordered as in the molecule input file `MOLECULE.INP`)

`.MAX IT`

```
READ (LUCMD,*),MAXESR
```

The line following gives the maximum number of iterations. (Default = 60)

`.PRINT`

```
READ (LUCMD,*),IPRESR
```

The line following gives the print level for the ESR routines.

`.THCESR`

```
READ (LUCMD,*),THCESR
```

The line following is the threshold for convergence (Default = 1.0D-5)

The following options are obsolete but are kept for backward compatibility. They are replaced by `.FCCALC` and `.SDCALC` above which also enables the printing of the tensors of the most important isotopes of the atoms in commonly used units.

```
.SNGPRP
    READ (LUCMD,'(A)'), LABEL
```
Singlet Operator. The line following is the label in the AOPROPER file.

```
.TRPPRP
    READ (LUCMD,'(A)'), LABEL
```
Triplet Operator. The line following is the label in the AOPROPER file.

Zero-field splitting: `.ZFS`

Calculation of the the electronic spin–spin contribution to the zero-field splitting tensor:

```
.ZFS
```

Electronic g-tensors: `.G-TENSOR`

Calculation of the electronic g-tensor:

```
.G-TENSOR
```
Initializes input block for g-tensor related options

The default is to calculate all contributions. The following options selects individual contributions

`.RMC` Relativistic mass correction

`.OZSO1` Second-order (paramagnetic) orbital-Zeeman + 1-electron spin-orbit contributions

`.OZSO2` Second-order (paramagnetic) orbital-Zeeman + 2-electron spin-orbit contributions

`.GC1` 1-electron gauge correction (diamagnetic) contributions

`.GC2` 2-electron gauge correction (diamagnetic) contributions

`.ECC` Choose electron center of charge (ECC) as gauge origin.

The following are utility options for modifying the default calculational procedure.

`.ADD-SO` Adds the 1- and 2-electron spin-orbit operators. This option may be used when one is not interested in the individual 1- and 2-electron contribution to the paramagnetic g-tensor, since it reduces the number of response equations to be solved.

`.MEAN-FIELD` Uses the approximate atomic mean field (AMFI) spin-orbit operator for evaluating the paramagnetic contributions.

.SCALED Uses the approximate 1-electron spin-orbit operator with scaled nuclear charges taken from Ref. [172] for evaluating the paramagnetic contributions.

.ZERO READ(LUCMD,'(A80)')G_LINE

This option is mainly for linear molecules in a Σ state. Specifying e.g. "ZZ" on the input line instructs the program to skip the calculation of the paramagnetic contribution to g_{zz}.

Chapter 27

Coupled-cluster calculations, CC

The coupled cluster program CC is designed for large-scale correlated calculations of energies and properties using a hierarchy of coupled cluster models: CCS, CC2, CCSD, and CC3, as well as standard methods such as MP2 and CCSD(T). The program offers almost the same options as the other parts of the Dalton program for SCF and MCSCF wave functions. It thus contains a wave function optimization section and a response function section where linear, quadratic and cubic response functions and electronic transition properties are calculated. For CC3 however, it is only possible to calculate the wave function and singlet excitation energies. At the moment, molecular gradients are available up to the CCSD(T) level [?] for ground states. London orbitals have so far not been implemented for the calculation of magnetic properties. In the manual, more details can be found about the specific implementations.

An additional feature of the program is that all levels of correlation treatment have been implemented using integral-direct techniques making it possible to run calculations using large basis sets [?].

In this chapter the general structure of the input for the coupled cluster program is described. The complete input for the coupled cluster program appears as sections in the input for the SIRIUS program, with the general input in the input section *CC INPUT. In order to get into the CC program one has to specify the .CC keyword in the general input section of SIRIUS **WAVE FUNCTIONS. For instance, a minimal input file for a CCSD(T) energy calculation would be:

```
**DALTON INPUT
.RUN WAVE FUNCTIONS
**WAVE FUNCTIONS
.CC
*CC INPUT
.CC(T)
```

```
**END OF DALTON INPUT
```

Several models can be calculated at the same time by specifying more models in the CC input section. The models supported in the CC program are CCS[173], MP2[174], CC2[173], the CIS(D) excitation energy approximation [175], CCSD[176], the CCSDR(3) excitation energy approximation[177], CCSD(T)[178], and CC3[179, 180]. Several electronic properties can also be calculated in one calculation by specifying simultaneously the various input sections in the *CC INPUT section as detailed in the following sections.

There is also a possibility for performing cavity coupled cluster self-consistent-reaction field calculations for solvent modelling [?].

27.1 General input for CC: *CC INPUT

In this section keywords for the coupled cluster program are defined. In particular the coupled cluster model(s) and other parameters common to all submodules are specified.

.BOSKIP Skip the calculation of the $B0$-vectors in a restarted run. See comments under .IMSKIP.

.CC(2) Run calculation for non-iterative CC2 excitation energy model (which is actually the well-known CIS(D) model).

.CC(3) Run calculation for CC(3) ground state energy model.

.CC(T) Run calculation for CCSD(T) (Coupled Cluster Singles and Doubles with perturbational treatment of triples) model.

.CC2 Run calculation for CC2 model.

.CC3 Run calculation for CC3 model.

.CCD Run calculation for Coupled Cluster Doubles (CCD) model.

.CIS Run calculation for CI Singles (CIS) method.

.CCR(3) Run calculation for CCSDR(3) excitation energy model.

.CCS Run calculation for Coupled Cluster Singles (CCS) model.

.CCSD Run calculation for Coupled Cluster Singles and Doubles (CCSD) model.

.CCSTST Test option which runs the CCS finite field calculation as a pseudo CC2 calculation. CCSTST disables all terms which depend on the double excitation amplitudes or multipliers. This flag must be set for CCS finite field calculations.

.DEBUG Test option: print additional debug output.

.EOSKIP Skip the calculation of the $E0$-vectors in a restarted run. See comment under .IMSKIP.

.F1SKIP Skip the calculation of F-matrix transformed first-order cluster amplitude responses in a restarted run. See comment under .IMSKIP.

.F2SKIP Skip the calculation of F-matrix transformed second-order cluster amplitude responses in a restarted run. See comment under .IMSKIP.

.FIELD

```
READ (LUCMD,*) EFIELD
READ (LUCMD,'(1X,A8)') LFIELD
```

Add an external finite field (operator label `LFIELD`) of strength `EFIELD` to the Hamiltonian (same input format as in Sec. 24.2.8). These fields are only included in the CC calculation and not in the calculation of the SCF reference state (*i.e.* the orbitals). Using this option in the calculation of numerical derivatives thus gives so-called orbital-unrelaxed energy derivatives, which is standard in coupled cluster response function theory. Note that this way of adding an external field does not work for models including triples excitations (most notably CCSD(T) and CC3). For finite field calculations of orbital-relaxed energy derivatives the field must be included in the SCF calculation. For such calculations, use the input section `*HAMILTONIAN` (Sec. 24.2.8).

.FREEZE

```
READ (LUCMD,*) NFC, NFV
```

Specify the number of frozen core orbitals and number of frozen virtuals (0 is recommended for the latter). The program will automatically freeze the NFC(NFV) orbitals of lowest (highest) orbital energy among the canonical Hartree–Fock orbitals. For benzene a relevant choice is for example 6 0.

.FROIMP

```
READ (LUCMD,*) (NRHFFR(I),I=1,MSYM)
READ (LUCMD,*) (NVIRFR(I),I=1,MSYM)
```

Specify for each irreducible representation how many orbitals should be frozen (deleted) for the coupled cluster calculation. In calculations, the first `NRHFFR(I)` orbitals will be kept frozen in symmetry class `I` and the last `NVIRFR(I)` orbitals will be deleted from the orbital list.

.FROEXP

```
READ (LUCMD,*) (NRHFFR(I),I=1,MSYM)
DO ISYM = 1, MSYM
  IF (NRHFFR(ISYM.NE.0) THEN
    READ (LUCMD,*) (KFRRHF(J,ISYM),J=1,NRHFFR(ISYM))
  END IF
END DO
READ (LUCMD,*) (NVIRFR(I),I=1,MSYM)
DO ISYM = 1, MSYM
  IF (NVIRFR(ISYM.NE.0) THEN
    READ (LUCMD,*) (KFRVIR(J,ISYM),J=1,NVIRFR(ISYM))
```

```
   END IF
END DO
```

Specify explicitly for each irreducible representation the orbitals that should be frozen (deleted) in the coupled cluster calculation.

.FRSKIP Skip the calculation of the F-matrix transformed right eigenvectors in a restarted run. See comment under `.IMSKIP`.

.HERDIR Run coupled cluster program AO-direct using HERMIT (Default is to run the program only AO integral direct if the `DIRECT` keyword was set in the general input section of DALTON. If `HERDIR` is not specified the ERI program is used as integral generator.)

.IMSKIP Skip the calculation of some response intermediates in a restarted run. This options and the following skip options is primarily for very experienced users and programmers who want to save a little bit of CPU time in restarts on very large calculations. *It is assumed that the user knows what he is doing when using these options, and the program does not test if the intermediates are there or are correct. The relevant quantities are assumed to be at the correct directory and files. These comments also applies for all the other skip options following!*

.L0SKIP Skip the calculation of the zeroth-order ground-state Lagrange multipliers in a restarted run. See comment under `.IMSKIP`.

.L1SKIP Skip the calculation of the first-order responses of the ground-state Lagrange multipliers in a restarted run. See comment under `.IMSKIP`.

.L2SKIP Skip the calculation of the second-order responses of the ground-state Lagrange multipliers in a restarted run. See comment under `.IMSKIP`.

.LESKIP Skip the calculation of left eigenvectors in a restarted run. See comment under `.IMSKIP`.

.LISKIP Skip the calculation of (ia|jb) integrals in a restarted run. See comment under `.IMSKIP`.

.M1SKIP Skip the calculation of the special zeroth-order Lagrange multipliers for ground-excited state transition moments, the so-called M-vectors, in a restarted run. See comment under `.IMSKIP`.

.MAX IT
```
READ (LUCMD,'(I5)') MAXITE
```
Maximum number of iterations for wave function optimization (default is `MAXITE = 40`).

.MAXRED

>   READ (LUCMD,*) MAXRED

>   Maximum dimension of the reduced space for the solution of linear equations (default is `MAXRED = 200`).

.MP2 Run calculation for second-order Møller-Plesset perturbation theory (MP2) method.

.MXDIIS

>   READ (LUCMD,*) MXDIIS

>   Maximum number of iterations for DIIS algorithm before wave function information is discarded and a new DIIS sequence is started (default is `MXDIIS = 8`).

.MXLRV

>   READ (LUCMD, *) MXLRV

>   Maximum number of trial vectors in the solution of linear equations. If the number of trial vectors reaches this value, all trial vectors and their transformations with the Jacobian are skipped and the iterative procedure for the solution of the linear (i.e. the response) equations is restarted from the current optimal solution.

.NOCCIT No iterations in the wave function optimization is carried out.

.NSIMLE

>   READ (LUCMD, *) NSIMLE

>   Set the maximum number of response equations that should be solved simultaneously. Default is 0 which means that all compatible equations (same equation type and symmetry class) are solved simultaneously.

.NSYM

>   READ (LUCMD,*) MSYM2

>   Set number of irreducibel representations.

.O2SKIP Skip the calculation of right-hand side vectors for the second-order cluster amplitude equations in a restarted run. See comment under `.IMSKIP`.

.PAIRS Decompose the singles and/or doubles coupled cluster correlation energy into contributions from singlet- and triplet-coupled pairs of occupied orbitals. Print those pair energies.

.PRINT

>   READ (LUCMD,'(I5)') IPRINT

Set print parameter for coupled cluster program (default is to take the value `IPRUSR`, set in the general input section of dalton).

.R1SKIP Skip the calculation of the first-order amplitude responses in a restarted run. See comment under `.IMSKIP`.

.R2SKIP Skip the calculation of the second-order cluster amplitude equations in a restarted run. See comment under `.IMSKIP`.

.RESKIP Skip the calculation of the first-order responses of the ground-state Lagrange multipliers in a restarted run. See comment under `.IMSKIP`.

.RESTART Try to restart the calculation from the cluster amplitudes, Lagrange multipliers, response amplitudes etc. stored on disk.

.SOPPA(CCSD) Write the CCSD singles and doubles amplitudes on the Sirius interface for later use in a SOPPA(CCSD) calculation. This chooses automatically the Coupled Cluster Singles and Doubles (CCSD) model.

.THRENR

> `READ (LUCMD,*) THRENR`

Set threshold for convergence of the ground state energy.

.THRLEQ

> `READ (LUCMD,*) THRENR`

Set threshold for convergence of the response equations.

.X2SKIP Skip the calculation of the $\eta^{(2)}$ intermediates (needed to build the right-hand side vectors for the second-order ground state Lagrange multiplier response equations) in a restarted run. See comment under `.IMSKIP`.

27.2 Ground state first-order properties: `*CCFOP`

In this Section, the keywords for the calculation of ground-state first-order (one-electron) properties are described. The calculation is evoked with the `*CCFOP` flag followed by the appropriate keywords as described in the list below. Note that `*CCFOP` assumes that the proper integrals are written on the AOPROPER file, and one therefore has to set the correct property integral keyword(s) in the `**INTEGRALS` input Section. For properties that have both an electronic and a nuclear contribution, these will be printed separately with a print level of 10 or above.

The calculation of first-order properties is implemented for the coupled cluster models CCS (which gives SCF first-order properties), CC2, MP2, CCD, CCSD and CCSD(T). By default, the chosen properties include orbital relaxation contributions, *i.e.* they are calculated from the relaxed CC (or MP) densities. To disable orbital relaxation for the CC2 and CCSD models, see `.NONREL` below. Relaxation is always included for MP2. Note also that the present implementation does not allow for CC2 relaxed first-order properties in the frozen core approximation (`.FREEZE` and `.FROIMP`, see Sec. 27.1).

For details on the implementation, see Refs. [181, 182, 183].

Reference literature:

A. Halkier, H. Koch, O. Christiansen, P. Jørgensen, and T. Helgaker
J. Chem. Phys., **107**, 849, (1997).

K. Hald, A. Halkier, P. Jørgensen, S. Coriani, C. Hättig, T. Helgaker
J. Chem . Phys., **118**, 2985, (2003) '

`.2ELDAR` Calculate relativistic two-electron Darwin term.

`.ALLONE` Calculate all first-order properties described in this Section (all corresponding property integrals are needed).

`.DIPMOM` Calculate the permanent molecular electric dipole moment (`DIPLEN` integrals).

`.NONREL` Compute the properties using the unrelaxed CC densities instead of the default relaxed densities.

`.NQCC` Calculate the electric field gradients at the nuclei (`EFGCAR` integrals).

`.OPERAT`
     READ (LUCMD,'(1X,A8)') LABPROP
     Calculate the electronic contribution to the property defined by the operator label `LABPROP` (corresponding `LABPROP` integrals needed).

.QUADRU Calculate the permanent traceless molecular electric quadrupole moment (`THETA` integrals). Note that the origin is the origin of the coordinate system specified in the MOLECULE.INP file.

.RELCOR Calculate scalar-relativistic one-electron corrections to the ground-state energy (`DARWIN` and `MASSVELO` integrals).

.SECMOM Calculate the electronic second moment of charge (`SECMOM` integrals).

.TSTDEN Calculate the CC energy using the two-electron CC density. Programmers keyword used for debugging purposes—Do not use.

27.3 Linear response functions: `*CCLR`

In the `*CCLR` section the input that is specific for coupled cluster linear response properties is read in. This section includes presently

- frequency-dependent linear response properties $\alpha_{AB}(\omega) = -\langle\langle A; B \rangle\rangle_\omega$ where A and B can be any of the one-electron operators for which integrals are available in the `**INTEGRALS` input part.

- dispersion coefficients $D_{AB}(n)$ for $\alpha_{AB}(\omega)$ which for $n \geq 0$ are defined by the expansion

$$\alpha_{AB}(\omega) = \sum_{n=0}^{\infty} \omega^n D_{AB}(n)$$

  In addition to the dispersion coefficients for $n \geq 0$ there are also coefficients available for $n = -1, \dots, -4$, which are related to the Cauchy moments by $D_{AB}(n) = S_{AB}(-n-2)$.
  Note, that for real response functions only even moments $D_{AB}(2n) = S_{AB}(-2n-2)$ with $n \geq -2$ are available, while for imaginary response functions only odd moments $D_{AB}(2n+1) = S_{AB}(-2n-3)$ with $n \geq -2$ are available.

Coupled cluster linear response functions and dispersion coefficients are implemented for the models CCS, CC2 and CCSD. The theoretical background for the implementation is detailed in Ref. [184, 185, 186]. The properties calculated are in the approach now generally known as coupled cluster linear response—in the frequency-independent limit this coincides with the so-called orbital-unrelaxed energy derivatives (and thus the orbital-unrelaxed finite field result).

Reference literature:

> Linear response: O. Christiansen, A. Halkier, H. Koch, P. Jørgensen, and T. Helgaker *J. Chem. Phys.*, **108**, 2801, (1998).

> Dispersion coefficients: C. Hättig, O. Christiansen, and P. Jørgensen *J. Chem. Phys.*, **107**, 10592, (1997).

`.ASYMSD` Use an asymmetric formulation of the linear response function which does not require the solution of response equations for the operators A, but solves two sets of response equations for the operators B.

`.AVERAG`
```
READ (LUCMD,'(A)') AVERAGE
READ (LUCMD,'(A)') SYMMETRY
```

Evaluate special tensor averages of linear response functions. Presently implemented are the isotropic average of the dipole polarizability $\bar{\alpha}$ and the dipole polarizability anisotropy α_{ani}. Specify ALPHA_ISO for AVERAGE to obtain $\bar{\alpha}$ and ALPHA_ANI to obtain α_{ani} and $\bar{\alpha}$. The SYMMETRY input defines the selection rules that can be exploited to reduce the number of tensor elements that have to be evaluated. Available options are ATOM, SPHTOP (spherical top), LINEAR, XYDEGN (x- and y-axis equivalent, *i.e.* a C_z^n symmetry axis with $n \geq 3$), and GENER (use point group symmetry from geometry input).

.DIPOLE Evaluate all symmetry allowed elements of the dipole polarizability (max. 6 components).

.DISPCF

```
READ (LUCMD,*) NLRDSPE
```

Calculate the dispersion coefficients $D_{AB}(n)$ up to $n = $ NLRDSPE.

.FREQUE

```
READ (LUCMD,*) NBLRFR
READ (LUCMD,*) (BLRFR(I),I=1,NBLRFR)
```

Frequency input for $\langle\langle A; B \rangle\rangle_\omega$.

.OPERAT

```
READ (LUCMD,'(2A)') LABELA, LABELB
DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
  READ (LUCMD,'(2A)') LABELA, LABELB
END DO
```

Read pairs of operator labels.

.PRINT

```
READ (LUCMD,*) IPRSOP
```

Set print level for linear response output.

27.4 Quadratic response functions: `*CCQR`

In the `*CCQR` section you specify the input for coupled cluster quadratic response calculations. This section includes:

- frequency-dependent third-order properties

$$\beta_{ABC}(\omega_A; \omega_B, \omega_C) = -\langle\langle A; B, C\rangle\rangle_{\omega_B, \omega_C} \qquad \text{with } \omega_A = -\omega_B - \omega_C$$

  where A, B and C can be any of the one-electron operators for which integrals are available in the `**INTEGRALS` input part.

- dispersion coefficients $D_{ABC}(n, m)$ for third-order properties, which for $n \geq 0$ are defined by the expansion

$$\beta_{ABC}(-\omega_B - \omega_C; \omega_B, \omega_C) = \sum_{n,m=0}^{\infty} \omega_B^n \, \omega_C^m \, D_{ABC}(n, m)$$

The coupled cluster quadratic response function is at present implemented for the coupled cluster models CCS, CC2 and CCSD.

The response functions are evaluated for a number of operator triples (given using the `.OPERAT`, `.DIPOLE`, or `.AVERAG` keywords) which are combined with pairs of frequency arguments specified using the keywords `.MIXFRE`, `.SHGFRE`, `.ORFREQ`, `.EOPEFR` or `.STATIC`. The different frequency keywords are compatible and might be arbitrarily combined or repeated. For dispersion coefficients use the keyword `.DISPCF`.

Reference literature:

Quadratic response: C. Hättig, O. Christiansen, H. Koch, and P. Jørgensen *Chem. Phys. Lett.*, **269**, 428, (1997).

Dispersion coefficients: C. Hättig, and P. Jørgensen *Theor. Chem. Acc.*, **100**, 230, (1998).

`.AVERAG`

    `READ (LUCMD,'(A)') LINE`

Evaluate special tensor averages of quadratic response properties. Presently implemented are only the vector averages of the first dipole hyperpolarizability $\beta_{||}$, $\beta_{\perp}$ and β_K. All three of these averages are obtained if `HYPERPOL` is specified on the input line that follows `.AVERAG`. The `.AVERAG` keyword should be used before any `.OPERAT` or `.DIPOLE` input in the `*CCQR` section.

`.DIPOLE` Evaluate all symmetry allowed elements of the first dipole hyperpolarizability (max. 27 components).

.DISPCF
```
READ (LUCMD,*) NQRDSPE
```
Calculate the dispersion coefficients $D_{ABC}(n,m)$ up to order $n + m =$NQRDSPE.

.EOPEFR
```
READ (LUCMD,*) MFREQ
READ (LUCMD,*) (BQRFR(IDX),IDX=NQRFREQ+1,NQRFREQ+MFREQ)
```
Input for the electro optical Pockels effect $\beta_{ABC}(-\omega; \omega, 0)$: on the first line following .EOPEFR the number of different frequencies is read, from the second line the input for $\omega_B = \omega$ is read. ω_C is set to 0 and ω_A to $\omega_A = -\omega$.

.MIXFRE
```
READ (LUCMD,*) MFREQ
READ (LUCMD,*) (BQRFR(IDX),IDX=NQRFREQ+1,NQRFREQ+MFREQ)
READ (LUCMD,*) (CQRFR(IDX),IDX=NQRFREQ+1,NQRFREQ+MFREQ)
```
Input for general frequency mixing $\beta_{ABC}(-\omega_B-\omega_C; \omega_B, \omega_C)$: on the first line following .MIXFRE the number of different frequencies (for this keyword) is read, from the second and third line the frequency arguments ω_B and ω_C are read (ω_A is set to $-\omega_B - \omega_C$).

.NOBMAT Test option: Do not use B matrix transformations but pseudo F matrix transformations (with the zeroth-order Lagrange multipliers exchanged by first-order responses) to compute the terms $\bar{t}^A \mathbf{B} t^B t^C$. This is usually less efficient.

.OPERAT
```
READ (LUCMD'(3A)') LABELA, LABELB, LABELC
DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
  READ (LUCMD'(3A)') LABELA, LABELB, LABELC
END DO
```
Read triples of operator labels. For each of these operator triples the quadratic response function will be evaluated at all frequency pairs. Operator triples which do not correspond to symmetry allowed combination will be ignored during the calculation.

.ORFREQ
```
READ (LUCMD,*) MFREQ
READ (LUCMD,*) (BQRFR(IDX),IDX=NQRFREQ+1,NQRFREQ+MFREQ)
```
Input for optical rectification $\beta_{ABC}(0; \omega, -\omega)$: on the first line following .ORFREQ the number of different frequencies is read, from the second line the input for $\omega_B = \omega$ is read. ω_C is set to $\omega_C = -\omega$ and ω_A to 0.

```
.PRINT
      READ (LUCMD,*) IPRINT
```

Set print parameter for the quadratic response section.

```
.SHGFRE
      READ (LUCMD,*) MFREQ
      READ (LUCMD,*) (BQRFR(IDX),IDX=NQRFREQ+1,NQRFREQ+MFREQ)
```

Input for second harmonic generation $\beta_{ABC}(-2\omega;\omega,\omega)$: on the first line following .SHGFRE the number of different frequencies is read, from the second line the input for $\omega_B = \omega$ is read. ω_C is set to ω and ω_A to -2ω.

.STATIC Add $\omega_A = \omega_B = \omega_C = 0$ to the frequency list.

.USE R2 Test option: use second-order response vectors instead of first-order Lagrange multiplier responses.

.XYDEGE Assume X and Y directions as degenerate in the calculation of the hyperpolarizability averages (this will prevent the program to use the components β_{zyy}, β_{yzy} β_{yyz} for the computation of the vector averages).

27.5 Cubic response functions: *CCCR

In the *CCCR section the input that is specific for coupled cluster cubic response properties is read in. This section includes:

- frequency-dependent fourth-order properties

$$\gamma_{ABCD}(\omega_A; \omega_B, \omega_C, \omega_D) = -\langle\langle A; B, C, D\rangle\rangle_{\omega_B, \omega_C, \omega_D} \quad \text{with } \omega_A = -\omega_B - \omega_C - \omega_D$$

where A, B, C and D can be any of the one-electron operators for which integrals are available in the **INTEGRALS input part.

- dispersion coefficients $D_{ABCD}(l, m, n)$ for $\gamma_{ABCD}(\omega_A; \omega_B, \omega_C, \omega_D)$ which for $n \geq 0$ are defined by the expansion

$$\gamma_{ABCD}(\omega_A; \omega_B, \omega_C, \omega_D) = \sum_{l,m,n=0}^{\infty} \omega_B^l \, \omega_C^m \, \omega_D^n D_{ABCD}(l, m, n)$$

Coupled cluster cubic response functions and dispersion coefficients are implemented for the models CCS, CC2 and CCSD.

The response functions are evaluated for a number of operator quadruples (specified with the keywords .OPERAT, .DIPOLE, or .AVERAG) which are combined with triples of frequency arguments specified using the keywords .MIXFRE, .THGFRE, .ESHGFR, .DFWMFR, .DCKERR, or .STATIC. The different frequency keywords are compatible and might be arbitrarely combined or repeated. For dispersion coefficients use the keyword .DISPCF.

> **Reference literature:**
>
> Cubic response: C. Hättig, O. Christiansen, and P. Jørgensen *Chem. Phys. Lett.*, **282**, 139, (1998).
>
> Dispersion coefficients: C. Hättig, and P. Jørgensen *Adv. Quantum Chem.*, **35**, 111, (1999).

.AVERAG

    READ (LUCMD,'(A)') AVERAGE
    READ (LUCMD,'(A)') SYMMETRY

Evaluate special tensor averages of cubic response functions. Presently implemented are the isotropic averages of the second dipole hyperpolarizability $\gamma_{||}$ and $\gamma_{\perp}$. Set AVERAGE to GAMMA_PAR to obtain $\gamma_{||}$ and to GAMMA_ISO to obtain $\gamma_{||}$ and $\gamma_{\perp}$. The SYMMETRY input defines the selection rules exploited to reduce the number of tensor elements that have to be evaluated. Available options are ATOM, SPHTOP (spherical

top), `LINEAR`, and `GENER` (use point group symmetry from geometry input). Note that the `.AVERAG` option should be specified in the `*CCCR` section before any `.OPERAT` or `.DIPOLE` input.

`.DCKERR`

> `READ (LUCMD,*) MFREQ`
> `READ (LUCMD,*) (DCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)`

Input for dc-Kerr effect $\gamma_{ABCD}(-\omega; 0, 0, \omega)$: on the first line following `.DCKERR` the number of different frequencies are read, from the second line the input for $\omega_D = \omega$ is read. ω_B and ω_C to 0 and ω_A to $-\omega$.

`.DFWMFR`

> `READ (LUCMD,*) MFREQ`
> `READ (LUCMD,*) (BCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)`

Input for degenerate four wave mixing $\gamma_{ABCD}(-\omega; \omega, \omega, -\omega)$: on the first line following `.DFWMFR` the number of different frequencies are read, from the second line the input for $\omega_B = \omega$ is read. ω_C is set to ω, ω_D and ω_A to $-\omega$.

`.DIPOLE` Evaluate all symmetry allowed elements of the second dipole hyperpolarizability (max. 81 components per frequency).

`.DISPCF`

> `READ (LUCMD,*) NCRDSPE`

Calculate the dispersion coefficients $D_{ABCD}(l, m, n)$ up to $l + m + n = $ `NCRDSPE`. Note that dispersion coefficients presently are only available for real fourth-order properties.

`.ESHGFR`

> `READ (LUCMD,*) MFREQ`
> `READ (LUCMD,*) (BCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)`

Input for electric field induced second harmonic generation $\gamma_{ABCD}(-2\omega; \omega, \omega, 0)$: on the first line following `.ESHGFR` the number of different frequencies are read, from the second line the input for $\omega_B = \omega$ is read. ω_C is set to ω, ω_D to 0 and ω_A to -2ω.

`.L2 BC` solve response equations for the second-order Lagrangian multipliers $\bar{t}^{BC}$ instead of the equations for the second-order amplitudes t^{AD}.

`.L2 BCD` solve response equations for the second-order Lagrangian multipliers $\bar{t}^{BC}$, $\bar{t}^{BD}$, $\bar{t}^{CD}$ instead of the equations for the second-order amplitudes t^{AD}, t^{AC}, t^{AB}.

```
.MIXFRE
     READ (LUCMD,*) MFREQ
     READ (LUCMD,*) (BCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)
     READ (LUCMD,*) (CCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)
     READ (LUCMD,*) (DCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)
```

Input for general frequency mixing $\gamma_{ABCD}(\omega_A; \omega_B, \omega_C, \omega_D)$: on the first line following .MIXFRE the number of differenct frequencies is read and from the next three lines the frequency arguments ω_B, ω_C, and ω_D are read (ω_A is set to $-\omega_B - \omega_C - \omega_D$).

.NO2NP1 test option: switch off $2n + 1$-rule for second-order Cauchy vector equations.

```
.OPERAT
     READ (LUCMD,'(4A)') LABELA, LABELB, LABELC, LABELD
     DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
       READ (LUCMD,'(4A)') LABELA, LABELB, LABELC, LABELD
     END DO
```

Read quadruples of operator labels. For each of these operator quadruples the cubic response function will be evaluated at all frequency triples. Operator quadruples which do not correspond to symmetry allowed combination will be ignored during the calculation.

```
.PRINT
     READ (LUCMD,*) IPRINT
```

Set print parameter for the cubic reponse section.

.STATIC Add $\omega_A = \omega_B = \omega_C = \omega_D = 0$ to the frequency list.

```
.THGFRE
     READ (LUCMD,*) MFREQ
     READ (LUCMD,*) (BCRFR(IDX),IDX=NCRFREQ+1,NCRFREQ+MFREQ)
```

Input for third harmonic generation $\gamma_{ABCD}(-3\omega; \omega, \omega, \omega)$: on the first line following .THGFRE the number of different frequencies is read, from the second line the input for $\omega_B = \omega$ is read. ω_C and ω_D are set to ω and ω_A to -3ω.

.USECHI test option: use second-order χ-vectors as intermediates

.USEXKS test option: use third-order ξ-vectors as intermediates

27.6 Calculation of excitation energies: `*CCEXCI`

In the `*CCEXCI` section the input that is specific for coupled cluster linear response calculation of electronic excitation energies is given. Coupled cluster linear response excitation energies are implemented for the iterative CC models CCS, CC2, CCSD, and CC3 for both singlet and triplet excited states. For singlet excited states the non-iterative models CC(2)(=CIS(D)) and CCSDR(3) are also available. For understanding the theoretical background for some aspects of the CC3 calculations consult also Ref. [187, 188, 179, 189].

Reference literature:

Singlet excitation energies: O. Christiansen, H. Koch, A. Halkier, P. Jørgensen, T. Helgaker, and A. M. Sanchez de Meras *J. Chem. Phys.*, **105**, 6921, (1996).

Triplet excitation energies: K. Hald, C. Hättig, and P. Jørgensen *J. Chem. Phys.*, **113**, 7765, (2000).

.CC2PIC Functions as CCSPIC but the picking is based on CC2 excitation energies.

.CCSDPI Functions as CCSPIC but the picking is based on CCSD excitation energies.

.CCSPIC

```
READ(LUCMD,*) OMPCCS
```

Keyword for picking a state with a given CCS excitation energy. Optimize a number of states in CCS with different symmetries and this option will pick the one in closest correspondence with the given input excitation energy, and skip the states at higher energies and other symmetries in the following calculation. Useful for example in numerical Hessian calculations on excited states.

.MARGIN Specifies the maximum allowed deviation of the actual excitation energy from the input excitation energy when using CCSPIC, CC2PIC and CCSDPI.

.NCCEXC

```
READ (LUCMD,*) (NCCEXCI(ISYM,1),ISYM=1,MSYM)
```

Give the number of states desired. For singlet states only, one single line is required with the number of excitation energies for each symmetry class (max. 8).
If also triplet states are desired an additional line is given in same format.

.NOSCOM For CC3 calculations only: indicates that no self-consistent solution should be sought in the partitioned CC3 algorithm.

.OMEINP

```
READ (LUCMD,*) (NOMINP(ISYM,1),ISYM=1,MSYM)
```

```
DO ISYM = 1, MSYM
  DO IOM = 1, NOMINP(ISYM,1)
      READ (LUCMD,*) IOMINP(IOM,ISYM,1),EOMINP(IOM,ISYM,1)
  ENDDO
ENDDO
```

A way to provide an input omega for the partitioned CC3 algorithm or restrict the self-consistent solution to specific states. If OMEINP is not specified the program uses the best choice available to it at that moment based on previous levels of approximations (CCSD or even better CCSDR(3)) and calculates all states as given by NCCEXC. IOMPINP is 1 for the lowest excited state of a given symmetry, 2 for the second lowest etc.

By giving an 0.0 input excitation energy (as EOMINP) the program takes the best previous approximation found in this run - otherwise the user can specify a qualified guess (perhaps from a previous calculation which is now restarted).

.THREXC The threshold for the solution of the excitation energies and corresponding response eigenvectors. The threshold is the norm of the residual for the eigenvalue equation. (Default: 1.0D-04).

.TOLSC For CC3 calculations only: Set tolerance for excitation energies for obtaining a self-consistent solution to the partitioned CC3 algorithm. Tolerance refers to the eigenvalue itself in the self-consistency iterations of the default solver for CC3. Not used in DIIS solver (see .R3DIIS). (Default: 1.0D-04).

.R3DIIS Use DIIS solver for CC3. This solver only scales linearly with the number of excited states in comparison to the default solver which scales quadratically. However this solver might fail in cases with states dominated by double or higher excited determinants. (Default: OFF).

27.7 Ground state–excited state transition moments: *CCLRSD

In the *CCLRSD section the input that is specific for coupled cluster response calculation of ground state–excited state electronic transition properties is read in. This section includes for example calculation of oscillator strength etc. The transition properties are implemented for the models CCS, CC2 and CCSD (singlet states only). The theoretical background for the implementation is detailed in Ref. [184, 185]. This section *CCLRSD has to be used in connection with *CCEXCI for the calculation of excited states.

Reference literature:

Ground-state transition moments: O. Christiansen, A. Halkier, H. Koch, P. Jørgensen, and T. Helgaker *J. Chem. Phys.*, **108**, 2801, (1998).

.DIPOLE Calculate the ground state–excited state dipole (length) transition properties including the oscillator strength.

.DIPVEL Calculate the ground state–excited state dipole-velocity transition properties including the oscillator strength in the dipole-velocity form. (The dipole length form is recommended for standard calculations).

.NO2N+1 Use an alternative, and normally less efficient, formulation for calculation the transition matrix elements (involving solution of response equations for all operators instead of solving for the so-called M vectors which is the default).

.OPERAT

    `READ (LUCMD,'(2A8)') LABELA, LABELB`

Read pairs of operator labels for which the residue of the linear response function is desired. Can be used to calculate the transition property for a given operator by specifying that operator twice. The operator can be any of the one-electron operators for which integrals are available in the **INTEGRALS input part.

.SELEXC

    `READ(LUCMD,*) IXSYM,IXST`

Select for which excited states the calculation of transition properties are carried. The default is all states according to the *CCEXCI input section (the program takes into account symmetry). For calculating selected states only, provide a list of symmetry and state numbers (order after increasing energy in each symmetry class). This list is read until next input label is found.

27.8 Ground state–excited state two-photon transition moments: `*CCTPA`

This section describes the calculation of two-photon absorption strengths through coupled cluster response theory. The two-photon transition strength is defined as

$$S_{AB,CD}^{of}(\omega) = \frac{1}{2}\{M_{of}^{AB}(-\omega)M_{fo}^{CD}(\omega) + [M_{of}^{CD}(-\omega)M_{fo}^{AB}(\omega)]^*\}$$

`*CCTPA` drives the calculation of the left ($M_{of}^{XY}(\omega)$) and right ($M_{fo}^{XY}(\omega)$) transition moments, and of the (diagonal) transition strengths $S_{AB,AB}^{of}(\omega)$. The methodology is implemented for the CCS, CC2, CCSD and CC3 models.

> **Reference literature:**
>
> C. Hättig, O. Christiansen, and P. Jørgensen *J. Chem. Phys.*, **108**, 8331, (1998).
>
> C. Hättig, O. Christiansen, and P. Jørgensen *J. Chem. Phys.*, **108**, 8355, (1998).

`.STATES`

```
    READ (LUCMD,'(A70)') LABHELP
    DO WHILE (LABHELP(1:1).NE.'.' .AND. LABHELP(1:1).NE.'*')
      READ (LUCMD,'(3A)') IXSYM, IXSTATE, SMFREQ
    END DO
```

Select one or more excited states (among those specified in `*CCEXCI`), and photon energies. The symmetry class (`IXSYM`) and the number a state f within that symmetry (`IXSTATE`) have to be given together with a photon energy ω `SMFREQ` (in atomic units). For each state and photon energy the moments $M_{of}^{AB}(-\omega)$ and $M_{fo}^{AB}(+\omega)$ and the strengths $S_{AB,AB}^{of}(\omega)$ are computed for all operator pairs. Instead of `.STATES` the equivalent keywords `.TRANSITION`, `.SELEXC`, or `.SELSTA` may be used.

`.HALFFR` Set the photon energies (frequencies) for the two-photon transition moments equal to half the excitation energy of the final state f as calculated with the present coupled cluster model. If this option is switched on, the photon energies given with the specification of the states (`.STATES`) will be ignored.

`.OPERAT`

```
    READ (LUCMD,'(4A)') LABELA, LABELB
    DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
      READ (LUCMD,'(4A)') LABELA, LABELB
    END DO
```

Read pairs of operator labels. For each pair the left and right two-photon transition moments and strengths $S^{of}_{AB,AB}(\omega)$ will be evaluated for all states and frequencies. Operator pairs which do not correspond to symmetry allowed combinations will be ignored during the calculation.

.DIPLEN Compute all symmetry-allowed elements of the dipole–dipole transition moment tensors in the length gauge and the corresponding transition strengths. In addition the three averages δ_F, δ_G, and δ_H are evaluated (see below).

.DIPVEL Compute all symmetry-allowed elements of the dipole–dipole transition moment tensors in the velocity gauge and the corresponding transition strengths.

.ANGMOM Compute all symmetry-allowed elements of the transition moment tensors with the operators A and B equal to a component of the angular momentum operator $\vec{l}$, which is proportional to the magnetic dipole operator.

.PRINT
    READ (LUCMD,*) IPRSM
    Read print level. Default is 0.

.USE X2 use the second-order vectors η^{AB} as intermediates. This may save some CPU time with the models CCS, CC2, and CCSD. It will be ignored in CC3 calculations, where it cannot be used.

.USE O2 use the second-order vectors ξ^{AB} as intermediates. This may save some CPU time with the models CCS, CC2, and CCSD. It will be ignored in CC3 calculations, where it cannot be used.

If no transitions have been specified using .STATES or one of the equivalent keywords, the default is to include all states specified in *CCEXCI with the photon energies set to half the excitation energies (*i.e.* the .HALFFR option is implied).

If the full dipole–dipole tensors (in length gauge) have been specified for the moments (*e.g.* using .DIPLEN), the following three isotropic averages will be evaluated:

$$
\begin{aligned}
\delta_F &= \frac{1}{30} \sum_{\alpha\beta=x,y,z} S^{0f}_{\alpha\alpha,\beta\beta}(\omega) \\
\delta_G &= \frac{1}{30} \sum_{\alpha\beta=x,y,z} S^{0f}_{\alpha\beta,\alpha\beta}(\omega) \\
\delta_H &= \frac{1}{30} \sum_{\alpha\beta=x,y,z} S^{0f}_{\alpha\beta,\beta\alpha}(\omega)
\end{aligned}
$$

The option .DIPLEN will be implied by default if no other operator pairs have been specified using one of the keywords .OPERAT, .DIPVEL, or .ANGMOM.

27.9 Ground state–excited state three-photon transition moments: `*CCTM`

This section describes the calculation of third-order transition moments and strengths. Three photon transition strengths are defined as

$$S_{ABC,DEF}^{of}(\omega_1,\omega_2) = \frac{1}{2}\{M_{of}^{ABC}(-\omega_1,-\omega_2)M_{fo}^{DEF}(\omega_1,\omega_2) + [M_{of}^{DEF}(-\omega_1,-\omega_2)M_{fo}^{ABC}(\omega_1,\omega_2)]^*\}$$

where $M_{of}^{ABC}(\omega_1,\omega_2)$ and $M_{fo}^{ABC}(\omega_1,\omega_2)$ are the left and right three photon transition moments, respectively. The methodology is implemented for the CC models CCS, CC2 and CCSD.

> **Reference literature:**
>
> C. Hättig, O. Christiansen, and P. Jørgensen *J. Chem. Phys.*, **108**, 8331, (1998).

`.DIPOLE`

Calculate the three-photon moments and strengths for all possible combinations of Cartesian components of the electric dipole moment operator (729 combinations).

`.OPERAT`

```
READ (LUCMD,'(6A)') LABELA, LABELB, LABELC, LABELD, LABELE, LABELF
DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
   READ (LUCMD,'(6A)') LABELA, LABELB, LABELC, LABELD, LABELE, LABELF
ENDDO
```

Select the sextuples of operator labels for which to calculate the three-photon transition moments. Operator sextuples which do not correspond to symmetry-allowed combinations will be ignored during the calculation.

`.PRINT`

```
READ (LUCMD,*) IPRTM
```

Read print level. Default is 0.

`.SELSTA`

```
READ (LUCMD,'(A70)') LABHELP
DO WHILE (LABHELP(1:1).NE.'.' .AND. LABHELP(1:1).NE.'*')
  READ (LUCMD,*) IXSYM, IXST, FREQB, FREQC
END DO
```

Select one or more excited states f (among those specified in `*CCEXCI`), and the laser frequencies. The symmetry (`IXSYM`) and state number (`IXST`) within that symmetry are then given, one pair (`IXSYM,IXST`) per line. `FREQB` and `FREQC` specify the laser frequencies ω_1 and ω_2 (in atomic units).

Default is all states specified in `*CCEXCI`, and for each state both laser frequencies equal to one third of the excitation energy.

`.THIRDF` Set the frequency arguments for the three-photon transition moments equal to one third of the excitation energy of the (chosen) final state f. Default, if `FREQB,FREQC` are not specified in `.SELSTA`.

27.10 Magnetic circular dichroism: *CCMCD

This section deals with the calculation of the $\mathcal{B}(o \to f)$ term of magnetic circular dichroism for a dipole-allowed transition between the ground state o and the excited state f. The $\mathcal{B}(o \to f)$ term results from an appropriate combination of products of electric dipole one-photon transition moments and mixed electric dipole-magnetic dipole two-photon transition moments— or, more precisely, a combination of transition strengths $S_{AB,C}^{of}(0.0)$— which are related to the single residues of the (linear and) quadratic response functions [17, 190, 191]:

$$\mathcal{B}(o \to f) \propto \varepsilon_{\alpha\beta\gamma} M_{of}^{\mu_\gamma m_\beta}(0.0) M_{fo}^{\mu_\alpha}$$

Within *CCMCD we specify the keywords needed in order to compute the individual contributions for each chosen transition $(o \to f)$ to the $\mathcal{B}(o \to f)$ term. *CCMCD has to be used in connection with *CCEXCI for the calculation of excited states. The calculation is implemented for the models CCS, CC2 and CCSD. Note that the output results refer to the magnetic moment operator.

> **Reference literature:**
>
> S. Coriani, C. Hättig, P. Jørgensen, and T. Helgaker *J. Chem. Phys.*, **113**, 3561, (2000).

.MCD All six triples of operators obtained from the permutations of 3 simultaneously different Cartesian components of the 3 operators `LABELA,LABELB,LABELC` are automatically specified. Operator triples which do not correspond to symmetry-allowed combinations will be ignored during the calculation. This is the default.

.OPERAT
```
READ (LUCMD,'(3A)') LABELA, LABELB, LABELC
DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
  READ (LUCMD,'(3A)') LABELA, LABELB, LABELC
END DO
```

Manually select the operator label triplets defining the operator Cartesian components involved in the $\mathcal{B}$ term. The first two operators (`LABELA,LABELB`) are those that enter the two-photon moment. The third operator (`LABELC`) is the one that enters the one-photon moment. They could be any of the (one-electron) operators for which integrals are available in **INTEGRALS. Specifically for the $\mathcal{B}$ term, `LABELA` corresponds to any component of the electric dipole moment operator, `LABELB` a component of the angular momentum (magnetic dipole) and `LABELC` to a component of the electric dipole. Operator triples which do not correspond to symmetry allowed combination will be ignored during the calculation.

```
.PRINT
      READ (LUCMD,*) IPRINT
```
Sets the print level in the *MCDCAL section. Default is IPRINT=0.

```
.SELSTA
      READ (LUCMD,'(A70)') LABHELP
      DO WHILE (LABHELP(1:1).NE.'.' .AND. LABHELP(1:1).NE.'*')
         READ(LABHELP,*) IXSYM,IXST
      END DO
```
Manually select one or more given excited states f among those specified in *CCEXCI. The symmetry (IXSYM) and state number (IXST) within that symmetry are then given, one pair (IXSYM,IXST) per line. Default is all dipole allowed states for all symmetries specified in *CCEXCI.

27.11 Transition moments between two excited states: `*CCQR2R`

In the `*CCQR2R` section the input that is specific for coupled cluster response calculation of excited state–excited state electronic transition properties is read in. This section includes presently for example calculation of excited state oscillator strength. The transition properties are available for the models CCS, CC2 and CCSD, but only for singlet states. The theoretical background for the implementation is detailed in Ref. [184, 185]. This section has to be used in connection with `*CCEXCI` for the calculation of excited states.

Reference literature:

O. Christiansen, A. Halkier, H. Koch, P. Jørgensen, and T. Helgaker
J. Chem. Phys., **108**, 2801, (1998).

`.DIPOLE` Calculate the ground state–excited state dipole (length) transition properties including the oscillator strength.

`.DIPVEL` Calculate the excited state–excited state dipole-velocity transition properties including the oscillator strength in the dipole-velocity form. (The dipole length form is recommended for standard calculations).

`.NO2N+1` Use an alternative, and normally less efficient, formulation for calculating the transition matrix elements (involing solution of response equations for all operators instead of solving for the so-called N vectors which is the default).

`.OPERAT`

        READ (LUCMD,'(2A8)') LABELA, LABELB

Read pairs of operator labels for which the residue of the linear response function is desired. Can be used to calculate the transition property for a given operator by specifying that operator twice. The operator can be any of the one-electron operators for which integrals are available in the `**INTEGRALS` input part.

`.SELEXC`

        READ(LUCMD,*) IXSYM1,IXST1,IXSYM2,IXST2

Select which excited states the calculation of transition properties are carried out for. The default is all states according to the CCEXCI input section (the program takes into account symmetry). For calculating selected states only, provide a list of symmetry and state numbers (order after increasing energy in each symmetry class) for state one and for state two. This list is read until next input label is found.

27.12 Excited-state first-order properties: *CCEXGR

In the *CCEXGR section the input that is specific for coupled cluster response calculation of excited-state first-order properties is read in. This section includes presently for example calculation of excited-state dipole moments and second moments of the electronic charge distribution. In many cases the information generated in this way is helpful for making qualitative assignments of the electronic states, for example in conjunction with the oscillator strengths and the orbital analysis of the response eigenvectors presented in the output.

The excited-state properties are available for the models CCS, CC2 and CCSD, but only for singlet states. The theoretical background for the implementation is detailed in Ref. [184, 185]. This section has to be used in connection with *CCEXCI for the calculation of excited states.

The properties calculated are in the approach now generally known as coupled cluster response—for these frequency independent properties this coincides with the so-called orbital-unrelaxed energy derivatives (and thus the orbital-unrelaxed finite-field result) for the excited-state total energies as obtained by the sum of the CC ground state energy and the CC response excitation energy.

Note of caution: Default in this section is therefore orbital *unrelaxed*, while for the ground state first order properties *CCFOP default is *relaxed*. To find results in a consistent approximation turn orbital relaxation off for the ground state (for CCS, CC2, CCSD) calculation.

> **Reference literature:**
>
> O. Christiansen, A. Halkier, H. Koch, P. Jørgensen, and T. Helgaker *J. Chem. Phys.*, **108**, 2801, (1998).

.ALLONE Calculate all of the above-mentioned excited state properties (all the above-mentioned property integrals are needed).

.DIPOLE Calculate the excited state permanent molecular electric dipole moment (DIPLEN integrals).

.NQCC Calculate the excited state electric field gradients at the nuclei (EFGCAR integrals).

.OPERAT

    READ (LUCMD,'(1X,A8)') LABPROP

Calculate the excited state electronic contribution to the property defined by the operator label LABPROP (corresponding LABPROP integrals needed).

.QUADRU Calculate the excited state permanent traceless molecular electric quadrupole moment (THETA integrals). Note that the origin is the origin of the coordinate system specified in the MOLECULE.INP file.

.RELCOR Calculate excited state scalar-relativistic one-electron corrections to the energy (`DARWIN` and `MASSVELO` integrals).

.SECMOM Calculate the excited state electronic second moment of charge (`SECMOM` integrals).

.SELEXC

> `READ(LUCMD,*) IXSYM,IXST`

> Select which excited states the calculation of excited state properties are carried out for. The default is all states according to the CCEXCI input section. When calculating selected states only, provide a list of symmetry and state numbers (order after increasing energy in each symmetry class). This list is read until next input label is found.

27.13 Excited state linear response functions and two-photon transition moments between two excited states: *CCEXLR

In the `*CCEXLR` section input that is specific for double residues of coupled cluster cubic response functions is read in. Results obtained using this functionality should cite Ref. [192, 193]. This section includes:

- frequency-dependent second-order properties of excited states

$$\alpha_{AB}^{(i)}(\omega) = -\langle\langle A; B\rangle\rangle_{\omega}^{(i)}$$

  where A and B can be any of the one-electron operators for which integrals are available in the `**INTEGRALS` input part.

- two-photon transition moments between excited states.

Double residues of coupled cluster cubic response functions are implemented for the models CCS, CC2, and CCSD.

Reference literature:

C. Hättig, and P. Jørgensen *J. Chem. Phys.*, **109**, 4745, (1998).

C. Hättig, O. Christiansen, S. Coriani, and P. Jørgensen *J. Chem. Phys.*, **109**, 9237, (1998).

`.ALLSTA` Calculate excited-state polarizabilities for all excited states.

`.DIPOLE` Evaluate all symmetry-allowed elements of the dipole–dipole tensor of the double residues of the cubic response function (a maximum of six components for second-order properties, and a maximum of nine for two-photon transition moments).

`.FREQ`
```
READ (LUCMD,*) MFREQ
READ (LUCMD,*) (BEXLRFR(IDX),IDX=NEXLRFR+1,NEXLRFR+MFREQ)
```
Frequency input for $\alpha_{AB}^{(i)}(\omega)$.

`.HALFFR` Use half the excitation energy as frequency argument for two-photon transition moments. Note that the `.HALFFR` keyword is incompatible with a user-specified list of frequencies.
For excited-state second-order properties the `.HALFFR` keyword is equivalent to the `.STATIC` keyword.

```
.OPERAT
     READ (LUCMD,'(2A)') LABELA, LABELB
     DO WHILE (LABELA(1:1).NE.'.' .AND. LABELA(1:1).NE.'*')
       READ (LUCMD,'(2A)') LABELA, LABELB
     END DO
```

Read pairs of operator labels. For each of these operator pairs, the double residues of the cubic response function will be evaluated at all frequencies. Operator pairs which do not correspond to symmetry-allowed combinations will be ignored during the calculation.

```
.PRINT
     READ (LUCMD,*) IPRINT
```

Set print parameter for the *CCEXLR section.

```
.SELSTA
     READ (LUCMD,'(A80)') LABHELP
     DO WHILE(LABHELP(1:1).NE.'.' .AND. LABHELP(1:1).NE.'*')
       READ(LUCMD,*) ISYMS(1), IDXS(1), ISYMS(2), IDXS(2)
     END DO
```

Read symmetry and index of the initial state and the final state. If initial and final state coincide one obtains excited state second-order properties (or, more precisely, the difference of the excited state second-order property relative to the ground-state property), if the two excited states are different one obtains the two-photon transition moments between the two excited states.

.STATIC Add $\omega = 0$ to the frequency list.

.USELEF Use left excited-state response vectors instead of the right excited-state response vectors (default is to use the right excited-state response vectors).

27.14 Numerical Gradients *CCGR

This section is used in the calculation of numerical derivatives of the CC energy. Since it is numerical it can be used for all models and both ground and excited states.

For excited states there is the problem of specifying which excited state is to be studied - and keeping track of this. One can specify the excited state by symmetry and number, using the keywords below. This works fine for gradients (though ordering may change in course of the optimization, but excited state optimization will inevitably be less black box than ground state optimizations), but can fail when there is symmetry lowering in the calculation of numerical hessians. For this purpose one can give the excitation energy for the appropriate state at a lower level (for example CCS) and from that find symmetry and number used in the real higher level calculation. It is implemented by the keywords in the CCEXCI section.

.NUMGD Specify that the calculation of gradients is to be done numerically. All that is required for coupled cluster (in addition to appropriate minimization, keywords see Chapter 7).

.XSTNUM
    READ (LUCMD,*) IXSTAT
    The number of the excited state for which the gradient is to be calculated (counted in terms of increasing energy).

.XSTSYM
    READ (LUCMD,*) IXSTSY
    Symmetry for excited state for which gradient is to be calculated.

27.15 R12 methods: `*R12`

The calculation of MP2-R12 energy corrections is requested. Note that an integral-direct calculation must be carried out and that the key `.R12` must be specified in the `**INTEGRALS` section.

> **Reference literature:**
>
> W. Klopper and C. C. M. Samson, *J. Chem. Phys.* **116**, 6397 (2002).
>
> C. C. M. Samson, W. Klopper and T. Helgaker, *Comp. Phys. Commun.* **149**, 1 (2002).

`.NO 1` Results for Ansatz 1 of the MP2-R12 method are not computed.

`.NO 2` Results for Ansatz 2 of the MP2-R12 method are not computed.

`.NO A` Results for approximation A of the MP2-R12 method are not printed.

`.NO A'` Results for approximation A′ of the MP2-R12 method are not printed.

`.NO B` Results for approximation B are not computed and not printed.

`.NO RXR` Those extra terms are ignored, which occur in approximation B when an auxiliary basis set is invoked for the resolution-of-identity (RI) approximation. If they are included, the results are marked as B′.

`.NO HYB` The default MP2-R12 calculation implemented in the Dalton program avoids two-electron integrals that involve two or more basis functions of the auxiliary basis (of course, only if such a basis is employed). This approach is denoted as hybrid scheme between approximations A and B. To obtain the full MP2-R12 energy in approximation B, the keyword `.NO HYB` must be specified. Then, two-electron integrals with up to two auxiliary basis functions are calculated but the calculation becomes more time-consuming.

`.R12DIA` The MP2-R12 equations are solved by diagonalizing the matrix representation of the Fock operator in the basis of R12 double replacements. If negative eigenvalues occur, a warning is issued. When this happens, the results should not be trusted, since the RI approximation appears to be insuffiently accurate. This diagonalizing is the default.

`.R12SVD` The MP2-R12 equations are solved by single value decomposition (the use of this keyword is not recommended).

.R12XXL All possible output from the MP2-R12 approach is generated (the use of this keyword is not recommended).

.SVDTHR
> `READ (LUCMD,*) SVDTHR`
> Threshold for singular value decomposition (default $= 10^{-12}$).

.VCLTHR
> `READ (LUCMD,*) SVDTHR`
> Threshold for neglect of R12 terms (default $= 0$, neglecting nothing).

Part IV

Appendix: DALTON Tool box

Appendix: DALTON Tool box

This appendix describes the pre- and postprocessing programs supplied to us by various users and authors. The programs are designed to create files directly useable by the DALTON program, or do various final analyses of one or more DALTON output files. We strongly encourage users to supply us with any such programs they have written in connection with their own use of the DALTON program, which we will be glad to distribute along with the DALTON program.

This appendix gives a short description of the programs supplied with the current distribution of the program, with proper references and a short description of the use of the program.

Executable versions of the programs in this directory will automatically be produced during installation of the DALTON program, but they will by default not be installed in the installation directory of the DALTON program, but rather placed in the `tools` directory.

A1: FChk2HES

Author: Gilbert Hangartner, Universität Freiburg, Switzerland

Purpose: Reads the Formatted Checkpoint file of Gaussian and creates a `DALTON.HES` file with Hessian and Atomic coordinates as well as a dummy `MOLECULE.INP` file.

Commandline: FChk2HES [-n] [filename]

-n does not read and write geometry; filename is optional, default is "`Test.FChk`" .

Comments: The Gaussian checkpointfile "Test.FChk" is created with the keyword "Form-Check=ForceCart", and where the keyword "NoSymm" has been specified

If symmetry is used: Hessian will be in standard(Gaussian) orientation, but geometry will be in input orientation. So do not use the `MOLECULE.INP` file from this utility, neither the coordinates in the end of `DALTON.HES`. Use either the standard-orientation from Gaussian outputfile, or use the converted checkpointfile (created with the command "form-chk checkpointfilename convertedfilename") as an input for this program. Anyway, this will not lead to the geometry you specified in the input, and the Hessian will be incompatible with the calculation you want to do in DALTON. (If this was already carried out at the input orientation ...)

The only way to get the geometry orientation you want is to turn off symmetry. Then, both the Hessian and the geometry in the Test.FChk file are in the input orientation, and everthing is fine.

The program has been made for the purpose of being used in VCD or VROA calculations where it may be of interest to compare predicted spectra obtained from SCF London orbital invariants/AATs with a MP2 or DFT force field.

A2: labread

Author: Hans Jørgen Aa.Jensen, Odense University, Denmark

Purpose: Reads unformatted AO-integral files and prints the INTEGRAL labels found on the file

Commandline: labread < infile > outfile

Comments: A convenient tool in connection with DALTON errors connected to missing labels on a given file in order to check that the given integrals do indeed, or do not, exist on a given file (usually `AOPROPER`).

A3: ODCPROG

Author: Antonio Rizzo (Istituto di Chimica Quantistica ed Energetica Molecolare, Pisa, Italy), and Sonia Coriani (University of Trieste, Italy)

Purpose: Analyze the magnetizability and nuclear shielding polarizabilities from a set of finite-field magnetizability and nuclear shielding calculations.

Commandline: odcprog

Requires the existence of a readmg.dat file, and n `DALTON.CM` files, where n is the number of finite field output files.

readmg.dat:

```
TITLE (1 line)
N ZPRT IPG ISSYM NIST NLAST NSTEP
FILE1
FILE2
...
FILEN
```

where

N Number of output DALTON.CM files for the FF calculations

ZPRT Sets the print level (T for maximum print level, F otherwise)

IPG Point group (1=Td, 2=Civ, 3=D2h, 4=C2v, 5=C3v, 6=Dih)

ISSYM Site symmetry of the atom for which the shielding polarizabilities are required

NIST First atom of which shielding polarizabilities must be computed (of those in the `DALTON.CM` files)

NLAST Last atom of which shielding polarizabilities must be computed (of those in the `DALTON.CM` files)

NSTEP Step to go from nist to nlast in the do-loop for shielding polarizabilities

FILEn nth `DALTON.CM` file (current name and location)

Comments: Only 6 point groups are presently implemented, of which C3v only for shielding-polarizabilities and Dih only for hyperpolarizabilities

When `issym.ne.ipg` and both hypermagnetizabilities and shielding polarizabilities are required, the number of field set-ups should be equal to the one for shieldings

Different field set-ups are needed according to molecular and/or nuclear site symmetries See for reference Raynes and Ratcliffe [194]. Linear molecules along the Z axis, planar molecules on XZ plane. In general, follow standard point symmetry arrangements

T_d Eg. CH_4 - No symmetry - 3 calculations

```
Need      0   0   0
Need      0   0   z
Need      x   0   z
```

$C_{\infty v}$ Eg. CO - No symmetry - 5 calculations

```
Need      0   0   0
Need      x   0   0
Need      0   0   z
Need      0   0   -z
Need      x   0   z
```

D_{2h} Eg. C_2H_4 - No symmetry - 7 calculations

```
Need      0   0   0
Need      0   0   z
Need      x   0   0
Need      0   y   0
Need      x   y   0
Need      x   0   z
Need      0   y   z
```

C_{2v} Eg. H_2O - No symmetry - 8 calculations

```
Need      0   0   0
Need      0   0   z
Need      0   0  -z
Need      x   0   0
Need      0   y   0
Need      x   y   0
Need      x   0   z
Need      0   y   z
```

C_{3v} Eq. Shielding H1 in CH_4

```
Need      0   0   0
Need      0   0   z
Need      0   0  -z
Need      x   0   0
Need     -x   0   0
Need      x   0   z
```

$D_{\infty h}$ Eg. N_2 - 4 calculations

```
Need      0   0   0
Need      0   0   z
Need      x   0   0
Need      x   0   z
```

Part V

References

Bibliography

[1] H. J. Aa. Jensen, H. Ågren, and J. Olsen. SIRIUS: a general-purpose direct second-order MCSCF program. In E. Clementi, editor, *Modern Techniques in Computational Chemistry*. ESCOM, Leiden, 1991.

[2] P. R. Taylor. Symmetry-adapted integral derivatives. *Theor. Chim. Acta*, 69:447, 1986.

[3] T. Yanai, D. P. Tew, and N. C. handy. *Chem. Phys. Lett.*, 393:51, 2004.

[4] Y. Tawada, T. Tsuneda, S. Yanagisawa, T. Yanai, and K. Hirao. *J. Chem. Phys.*, 120:8425, 2004.

[5] C. Angeli, R. Cimiraglia, S. Evangelisti, T. Leininger, and J.-P. Malrieu. *J. Chem. Phys.*, 114:10252, 2001.

[6] C. Angeli, R. Cimiraglia, and J.-P. Malrieu. *Chem. Phys. Lett.*, 350:297, 2001.

[7] C. Angeli, R. Cimiraglia, and J.-P. Malrieu. *J. Chem. Phys.*, 117:9138, 2002.

[8] K. Andersson, P.-Å. Malmqvist, and B. O. Roos. *J. Chem. Phys.*, 96:1218, 1992.

[9] K. G. Dyall. *J. Chem. Phys.*, 102:4909, 1995.

[10] W. Klopper and C. C. M. Samson. *J. Chem. Phys.*, 116:6397, 2002.

[11] C. C. M. Samson, W. Klopper, and T. Helgaker. *Comp. Phys. Commun.*, 149:1, 2002.

[12] P. Norman, D. M. Bishop, H. J. Aa. Jensen, and J. Oddershede. *J. Chem. Phys.*, 115:10323, 2001.

[13] F. Jensen. *J. Chem. Phys.*, 115:9113, 2001.

[14] P.-O. Åstrand, K. Ruud, and P. R. Taylor. Calculation of the vibrational wave function of polyatomic molecules. *J. Chem. Phys.*, 112:2655, 2000.

[15] K. Ruud, P.-O. Åstrand, and P. R. Taylor. An efficient approach for calculating vibrational wave functions and zero-point vibrational corrections to molecular properites of polyatomic molecules. *J. Chem. Phys.*, 112:2668, 2000.

[16] K. Ruud, J. Lounila, and J. Vaara. Temperature- and isotope-effects on nuclear magnetic shielding constants in polyatomic molecules. To be submitted.

[17] S. Coriani, P. Jørgensen, A. Rizzo, K. Ruud, and J. Olsen. *Ab initio* determinations of Magnetic Circular Dichroism. *Chem. Phys. Lett.*, 300:61, 1999.

[18] P.-O. Widmark, P.-Å. Malmqvist, and B. O. Roos. Density matrix averaged atomic natural orbital (ANO) basis sets for correlated molecular wave functions. *Theor. Chim. Acta*, 77:291, 1990.

[19] P.-O. Widmark, B. J. Persson, and B. O. Roos. *Theor. Chim. Acta*, 79:419, 1991.

[20] H. J. Aa. Jensen, P. Jørgensen, H. Ågren, and J. Olsen. Second-order Møller–Plesset perturbation theory as a configuration and orbital generator in multiconfiguration self-consistent field calculations. *J. Chem. Phys.*, 88:3834, 1988. Erratum **89**, 5354 (1988).

[21] H. J. Aa. Jensen, P. Jørgensen, and H. Ågren. Efficient optimization of large-scale MCSCF wave-functions with a restricted step algorithm. *J. Chem. Phys.*, 87:451, 1987.

[22] T. U. Helgaker, J. Almlöf, H. J. Aa. Jensen, and P. Jørgensen. Molecular Hessians for large-scale MCSCF wave functions. *J. Chem. Phys.*, 84:6266, 1986.

[23] R. Fletcher. *Practical Methods of Optimization Vol.1 - Unconstrained Optimization*. J. Wiley & Sons Ltd., New York, 1981.

[24] T. Helgaker. Transition-state optimizations by trust-region image minimization. *Chem. Phys. Lett.*, 182:503, 1991.

[25] P. Jørgensen, H. J. Aa. Jensen, and T. Helgaker. A gradient extremal walking algorithm. *Theor. Chim. Acta*, 73:55, 1988.

[26] H. J. Aa. Jensen, P. Jørgensen, and T. Helgaker. Systematic determination of MCSCF equilibrium and transition structures and reaction paths. *J. Chem. Phys.*, 85:3917, 1986.

[27] A. Banerjee, N. Adams, J. Simons, and R. Shephard. *J. Phys. Chem.*, 89:52, 1985.

[28] T. Helgaker, E. Uggerud, and H. J. Aa. Jensen. Integration of the classical equations of motion on *ab initio* molecular potential energy surfaces using gradients and Hessians: application to translational energy release upon fragmentation. *Chem. Phys. Lett.*, 173:145, 1990.

[29] R. Steckler, G. M. Thurman, J. D. Watts, and R. J. Bartlett. *Ab initio* direct dynamics study of OH + HCL → Cl + H_2O. *J. Chem. Phys.*, 106:3926, 1997.

[30] V. Bakken and T. Helgaker. *J. Chem. Phys.*, 117:9160, 2002.

[31] G. Fogarasi, X. F. Zhou, P. W. Taylor, and P. Pulay. The calculation of *ab initio* molecular geometries – efficient optimization by natural internal coordinates and empirical correction by offset forces. *J. Am. Chem. Soc.*, 114:8191, 1992.

[32] P. Pulay and G. Fogarasi. Geometry optimization in redundant internal coordinates. *J. Chem. Phys.*, 96:2856, 1992.

[33] R. Lindh, A. Bernhardsson, G. Karlström, and P.-Å. Malmqvist. On the use of a Hessian model function in molecular geometry optimizations. *Chem. Phys. Lett.*, 241:423, 1995.

[34] J. Baker. Techniques for geometry optimization: A comparison of Cartesian and natural internal coordinates. *J. Comput. Chem.*, 14:1085, 1993.

[35] J. M. Bofill. Updated Hessian matrix and the restricted step method for locating transition structures. *J. Comput. Chem.*, 15:1, 1994.

[36] C. J. Cerjan and W. H. Miller. On finding transition states. *J. Chem. Phys.*, 75:2800, 1981.

[37] K. Fukui. *Acc. Chem. Res.*, 14:363, 1981.

[38] K. Ruud, T. Helgaker, and E. Uggerud. *J. Mol. Struct. (THEOCHEM)*, 393:59, 1997.

[39] J. Cioslowski. A new population analysis based on atomic polar tensors. *J. Am. Chem. Soc.*, 111:8333, 1989.

[40] P.-O. Åstrand, K. Ruud, K. V. Mikkelsen, and T. Helgaker. Atomic charges of the water molecule and the water dimer. *J. Phys. Chem.* **A**, 102:7686, 1998.

[41] G. Placzek. *Handbuch der Radiologie*, 6(2):205, 1934.

[42] T. Helgaker, K. Ruud, K. L. Bak, P. Jørgensen, and J. Olsen. Vibrational raman optical activity calculations using London atomic orbitals. *Faraday Discuss.*, 99:165, 1994.

[43] A. D. Buckingham. Permanent and induced molecular moments and long-range intermolecular forces. *Adv. Chem. Phys.*, 12:107, 1967.

[44] W. Hüttner, M. K. Lo, and W. H. Flygare. Molecular g-value tensor, the molecular susceptibility tensor, and the sign of the electric dipole moment in formaldehyde. *J. Chem. Phys.*, 48:1206, 1968.

[45] J. Spiekermann and D. H. Sutter. Molecular g-values, magnetic susceptibility anisotropies, second moments of the electronic charge distribution , molecular electric quadrupole moment, and ^{14}N nuclear quadrupole coupling of nitroethylene, CH_2CHNO_2. *Z. Naturforsch.* **A**, 46:715, 1991.

[46] K. Ruud and T. Helgaker. The magnetizability, rotational g tensor, and quadrupole moment of PF_3 revisited. *Chem. Phys. Lett.*, 264:17, 1997.

[47] M. Jaszuński, S. Szymánski, O. Christiansen, P. Jørgensen, T. Helgaker, and K. Ruud. NMR properties of N_3^- – a comparison of theory and experiment. *Chem. Phys. Lett.*, 243:144, 1995.

[48] J. Olsen and P. Jørgensen. Linear and nonlinear response functions for an exact state and for an MCSCF state. *J. Chem. Phys.*, 82:3235, 1985.

[49] E. S. Nielsen, P. Jørgensen, and J. Oddershede. *J. Chem. Phys.*, 73:6238, 1980.

[50] J. Oddershede, P. Jørgensen, and D. Yeager. *Comput. Phys. Rep.*, 2:33, 1984.

[51] M. J. Packer, E. K. Dalskov, T. Enevoldsen, H. J. Aa. Jensen, and J. Oddershede. J. Chem. Phys., submitted.

[52] E. K. Dalskov and S. P. A.Sauer. *J. Phys. Chem.* **A**, 102:5269, 1998.

[53] S. P. A. Sauer. Second order polarization propagator approximation with cupled cluster singles and doubles amplitudes - soppa(ccsd): The polariability and hyperpolarizability of li. *J. Phys. B: At. Mol. Phys.*, 30:3773–3780, 1997.

[54] K. Wolinski, J. F. Hinton, and P. Pulay. Efficient implementation of the gauge-independent atomic orbital method for NMR chemical shift calculations. *J. Am. Chem. Soc.*, 112:8251, 1990.

[55] K. Ruud, T. Helgaker, R. Kobayashi, P. Jørgensen, K. L. Bak, and H. J. Aa. Jensen. Multiconfigurational self consistent field (MCSCF) calculations of nuclear shieldings using London atomic orbitals. *J. Chem. Phys.*, 100:8178, 1994.

[56] K. Ruud, T. Helgaker, K. L. Bak, P. Jørgensen, and H. J. Aa. Jensen. Hartree–Fock limit magnetizabilities from London orbitals. *J. Chem. Phys.*, 99:3847, 1993.

[57] K. Ruud, T. Helgaker, K. L. Bak, P. Jørgensen, and J. Olsen. Accurate magnetizabilities of the isoelectronic series BeH^-, BH and CH^+. the MCSCF-GIAO approach. *Chem. Phys.*, 195:157, 1995.

[58] P. Lazzeretti, M. Malagoli, and R. Zanasi. Computational approach to molecular magnetic properties by continuous transformation of the origin of the current density. *Chem. Phys. Lett.*, 220:299, 1994.

[59] P. Lazzeretti. Ring currents. *Prog. Nuc. Magn. Prop. Spectr*, 36:1–88, 2000.

[60] A. Ligabue, S. P. A. Sauer, and P. Lazzeretti. Correlated and gauge inveariant calculations of nuclear magnetic shieldings constants using the CTOCD-DZ approach. *J. Chem. Phys.*, 118:6830–6845, 2003.

[61] O. Vahtras, H. Ågren, P. Jørgensen, H. J. Aa. Jensen, S. B. Padkjær, and T. Helgaker. Indirect nuclear spin-spin coupling constants from multiconfigurational linear response theory. *J. Chem. Phys.*, 96:6120, 1992.

[62] S. P. A. Sauer. Theoretical estimates of the rotational g-factor, magnetizability and electric dipole moment of GaH. *Chem. Phys. Lett.*, 260:271, 1996.

[63] T. Enevoldsen, J. Oddershede, and S. P. A. Sauer. Correlated calculations of indirect nuclear spin-spin coupling constants using second-order polarization propagator approxiamtions: SOPPA and SOPPA(CCSD). *Theor. Chem. Acc.*, 100:275, 1998.

[64] J. Olsen, K. L. Bak, K. Ruud, T. Helgaker, and P. Jørgensen. Orbital connections for perturbation dependent basis sets. *Theor. Chim. Acta*, 90:421, 1995.

[65] T. H. Dunning Jr. Gaussian basis sets for use in correlated molecular calculations. I. The atoms boron through neon and hydrogen. *J. Chem. Phys.*, 90:1007, 1989.

[66] R. A. Kendall, T. H. Dunning Jr., and R. J. Harrison. Electron-affinities of the 1st-row atoms revisited – systematic basis-sets and wave-functions. *J. Chem. Phys.*, 96:6796, 1992.

[67] D. E. Woon and T. H. Dunning Jr. Gaussian basis sets for use in correlated molecular calculations. III. The atoms aluminium through argon. *J. Chem. Phys.*, 98:1358, 1993.

[68] D. E. Woon and T. H. Dunning Jr. Gaussian basis sets for use in correlated molecular calculations. IV. Calculation of static electrical response properties. *J. Chem. Phys.*, 100:2975, 1994.

[69] K. Ruud, T. Helgaker, P. Jørgensen, and K. L. Bak. Theoretical calculations of the magnetizability of some small fluorine-containing molecules using London atomic orbitals. *Chem. Phys. Lett.*, 223:12, 1994.

[70] K. Ruud, H. Skaane, T. Helgaker, K. L. Bak, and P. Jørgensen. Magnetizability of hydrocarbons. *J. Am. Chem. Soc.*, 116:10135, 1994.

[71] A. Schäfer, H. Horn, and R. Ahlrichs. Fully optimized contracted Gaussian basis sets for atoms Li to Kr. *J. Chem. Phys.*, 97:2571, 1992.

[72] A. Schäfer, C. Huber, and R. Ahlrichs. *J. Chem. Phys.*, 100:5829, 1994.

[73] T. Helgaker, M. Jaszuński, and K. Ruud. *Ab Initio* methods for the calculation of NMR shielding constants and indirect spin-spin coupling constants. *Chem. Rev.*, 99:293, 1999.

[74] J. Gauss, K. Ruud, and T. Helgaker. Perturbation-dependent atomic orbitals for the calculation of spin-rotation constants and rotational g tensors. *J. Chem. Phys.*, 105:2804, 1996.

[75] K. Ruud, T. Helgaker, P. Jørgensen, and K. L. Bak. An *ab initio* nuclear magnetic resonance spectrum of vinyllithium. *Chem. Phys. Lett.*, 226:1, 1994.

[76] B. Fernandez, P. Jørgensen, J. Byberg, J. Olsen, T. Helgaker, and H. J. Aa. Jensen. Spin polarization in restricted electronic-structure theory – multiconfiguration self-consistent-field calculations of hyperfine coupling-constants. *J. Chem. Phys.*, 97:3412, 1992.

[77] B. Fernandez and P. Jørgensen. Evaluation of hyperfine coupling tensors of the BeH and BeF radicals. *Chem. Phys. Lett.*, 232:463, 1995.

[78] K. L. Bak, P. Jørgensen, T. Helgaker, K. Ruud, and H. J. Aa. Jensen. Gauge-origin independent multiconfigurational self-consistent-field theory for vibrational circular dichroism. *J. Chem. Phys.*, 98:8873, 1993.

[79] K. L. Bak, Aa. E. Hansen, K. Ruud, T. Helgaker, J. Olsen, and P. Jørgensen. *Ab initio* calculation of electronic circular dichroism for trans-cyclooctene using London atomic orbitals. *Theor. Chim. Acta*, 90:441, 1995.

[80] P. L. Polavarapu. *Ab initio* molecular optical rotations and absolute configurations. *Mol. Phys.*, 91:551, 1997.

[81] P. L. Polavarapu, D. K.Chakraborty, and K. Ruud. Molecular optical rotations: an evaluation of semiempirical models. *Chem. Phys. Lett.*, 319:595, 2000.

[82] K. Ruud, T. Helgaker, J. Olsen, P. Jørgensen, and K. L. Bak. A numerically stable orbital connection for the calculation of analytical Hessians using perturbation dependent basis sets. *Chem. Phys. Lett.*, 235:47, 1995.

[83] K. L. Bak, P. Jørgensen, T. Helgaker, K. Ruud, and H. J. Aa. Jensen. Basis set convergence of atomic axial tensors obtained from self-consistent field calculations using London atomic orbitals. *J. Chem. Phys.*, 100:6620, 1994.

[84] K. L. Bak, P. Jørgensen, T. Helgaker, and K. Ruud. Basis set convergence and correlation effects in vibrational circular dichroism calculations using London atomic orbitals. *Faraday Discuss.*, 99:121, 1994.

[85] M. Pecul, K. Ruud, and T. Helgaker. *Chem. Phys. Lett.*, 388:110, 2004.

[86] J. R. Cheeseman, M. J. Frisch, F. J. Devlin, and P. J. Stephens. Hartree–Fock and density functional theory *ab initio* calculation of optical rotation using GIAOs: Basis set dependence. *J. Phys. Chem.* **A**, 104:1039, 2000.

[87] R. D. Amos. Electric and magnetic properties of CO, HF, and HCl, and CH_3F. *Chem. Phys. Lett.*, 87:23, 1982.

[88] G. Zuber and W. Hug. *J. Phys. Chem.* **A**, 108:2108, 2004.

[89] K. Ruud, D. Jonsson, P. Norman, H. Ågren, T. Saue, H. J. Aa. Jensen, P. Dahle, and T. Helgaker. Generalized integral-screening for efficient calculations of nonlinear optical properties of large molecules. *J. Chem. Phys.*, 108:7973, 1998.

[90] P. Norman, D. Jonsson, H. Ågren, P. Dahle, K. Ruud, T. Helgaker, and H. Koch. Efficient parallel implementation of response theory: Calculations of the second hyperpolarizability of polyacenes. *Chem. Phys. Lett.*, 253:1, 1996.

[91] M. Pecul, T. Saue, K. Ruud, and A. Rizzo. *J. Chem. Phys.*, 121:xxxx, 2004.

[92] A. Rizzo, T. Helgaker, K. Ruud, A. Barszczewicz, M. Jaszuński, and P. Jørgensen. Electric field dependence of magnetic properties: Multiconfigurational self-consistent field calculations of hypermagnetizabilities and shielding polarizabilities. *J. Chem. Phys.*, 102:8953, 1995.

[93] K. V. Mikkelsen, E. Dalgaard, and P. Swanstrøm. Electron-transfer reactions in solution – An *ab initio* approach. *J. Phys. Chem.*, 91:3081, 1987.

[94] K. V. Mikkelsen, H. Ågren, H. J. Aa. Jensen, and T. Helgaker. A multiconfigurational self-consistent reaction-field method. *J. Chem. Phys.*, 89:3086, 1988.

[95] K. V. Mikkelsen, P. Jørgensen, and H. J. Aa. Jensen. A multiconfigurational self-consistent reaction field response method. *J. Chem. Phys.*, 100:6597, 1994.

[96] K. V. Mikkelsen, Y. Luo, H. Ågren, and P. Jørgensen. Solvent induced polarizabilities and hyperpolarizabilities of para-nitroaniline studied by reaction field linear response theory. *J. Chem. Phys.*, 100:8240, 1994.

[97] B. Fernandez, O. Christiansen, O. Bludsky, P. Jørgensen, and K. V. Mikkelsen. Theory of hyperfine coupling constants of solvated molecules: Applications involving methyl and ClO$_2$ radicals in different solvents. *J. Chem. Phys.*, 104:629, 1996.

[98] K. V. Mikkelsen, A. Cesar, H. Ågren, and H. J. Aa. Jensen. Multiconfigurational self-consistent reaction field-theory for nonequilibrium solvation. *J. Chem. Phys.*, 103:9010, 1995.

[99] K. V. Mikkelsen, P. Jørgensen, K. Ruud, and T. Helgaker. A multipole reaction field model for gauge origin independent magnetic properties of solvated molecules. *J. Chem. Phys.*, 106:1170, 1997.

[100] P.-O. Åstrand, K. V. Mikkelsen, K. Ruud, and T. Helgaker. Magnetizabilities and nuclear shielding constants of the fluoromethanes in gas phase and solution. *J. Phys. Chem.*, 100:19771, 1996.

[101] K. Ruud, P.-O. Åstrand, and P. R. Taylor. Zero-point vibrational effects on proton shieldings: Functional-group contributions from *ab initio* calculations. *J. Am. Chem. Soc.*, 123:4826, 2001.

[102] R. M. Pitzer and N. M. Pitzer. *Int. J. Quantum Chem.*, 40:773, 1991.

[103] B. A. Hess, C. M. Marian, U. Wahlgren, and O. Gropen. *Chem. Phys. Lett.*, 251:365, 1996.

[104] C. Angeli, R. Cimiraglia, S. Evangelisti, T. Leininger, and J. P. Malrieu. Introduction of n–electron valence states for multireference perturbation theory. *J. Chem. Phys.*, 114:10252, 2001.

[105] C. Angeli, R. Cimiraglia, and J. P. Malrieu. n–electron valence state perturbation theory: a fast implementation of the strongly contracted variant. *Chem. Phys. Lett.*, 350:297, 2001.

[106] C. Angeli, R. Cimiraglia, and J. P. Malrieu. n–electron valence state perturbation theory: a spinless formulation and an efficient implementation of the strongly contracted and of the partially contracted variants. *J. Chem. Phys.*, 117:9138, 2002.

[107] C. Angeli, S. Borini, and R. Cimiraglia. An application of second-order n-electron valence state perturbation theory (nevpt2) to the calculation of excited states. *Theor. Chem. Acc.*, submitted.

[108] K. Ruud T. Helgaker and P. R. Taylor. Second-order methods for the optimization of molecular potential energy surfaces. In D. Heidrich, editor, *The reaction path in Chemistry: Current Approaches and Perspectives*. Kluwer, Netherlands, 1995.

[109] 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.

[110] A. J. Robinson. Vrml, chemistry and the web – a new reality. *Chem. Design Automat. News*, 10:50, 1995.

[111] P. Csàszàr and P. Pulay. Geometry optimization by direct inversion in the iterative subspace. *J. Mol. Struct.*, 114:31, 1984.

[112] H. B. Schlegel. Optimization of equilibrium geometries and transition structures. *J. Comput. Chem.*, 3:214, 1982.

[113] T. A. Ruden, P. R. Taylor, and T. Helgaker. Automated calculation of fundamental frequencies: application to AlH_3 using the CCSD(T) method. *J. Chem. Phys.*, 119:1951–1960, 2003.

[114] P. E. M. Siegbahn, J. Almlöf, A. 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, 1981.

[115] T. Helgaker and P. Jørgensen. An electronic Hamiltonian for origin independent calculations of magnetic properties. *J. Chem. Phys.*, 95:2595, 1991.

[116] S. Kirpekar, J. Oddershede, and H. J. Aa. Jensen. Relativistic corrections to molecular dynamic dipole polarizabilities. *J. Chem. Phys.*, 103:2983, 1995.

[117] D. Matsuoka and T. Aoyama. Molecular integral of diamagnetic contribution to nuclear spin–spin coupling constant. *J. Chem. Phys.*, 73:5718, 1980.

[118] K. L. Bak, P. Jørgensen, H. J. Aa. Jensen, J. Olsen, and T. Helgaker. First-order nonadiabatic coupling matrix elements from multiconfigurational self-consistent-field response theory. *J. Chem. Phys.*, 97:7573, 1992.

[119] B. A. Hess. *Phys.Rev.* **A**, 33:3742, 1986.

[120] T. B. Pedersen, H. Koch, and K. Ruud. Coupled cluster response calculation of natural chiroptical spectra. *J. Chem. Phys.*, 110:2883, 1999.

[121] J. Vaara, K. Ruud, and O. Vahtras. Second- and third-order spin-orbit contributions to nuclear shielding tensors. *J. Chem. Phys.*, 111:2900, 1999.

[122] J. Vaara, K. Ruud, and O. Vahtras. Correlated response calculations of the spin-orbit interaction contribution to nuclear spin-spin couplings. *J. Comput. Chem.*, 20:1314, 1999.

[123] O. Vahtras, H. Ågren, P. Jørgensen, H. J. Aa. Jensen, T. Helgaker, and J. Olsen. Spin-orbit coupling constants in a multiconfigurational linear response approach. *J. Chem. Phys.*, 96:2118, 1992.

[124] K. Kaufman, W. Baumeister, and M. Jungen. *J. Phys.* **B***, Mol. Opt. Phys*, 22:2223, 1989.

[125] I. Panas. Aspects of density functional theory in *ab initio* quantum chemistry: external correlation for free. *Chem. Phys. Lett.*, 245:171, 1995.

[126] J. Almlöf. A vectorized Gaussian integral program.

[127] Basis sets were obtained from the Extensible Computational Chemistry Environment Basis Set Database, Version 1.0, as developed and distributed by the Molecular Science Computing Facility, Environmental and Molecular Sciences Laboratory which is part of the Pacific Northwest Laboratory, P.O. Box 999, Richland, Washington 99352, USA, and funded by the U.S. Department of Energy. The Pacific Northwest Laboratory is a multi-program laboratory operated by Battelle Memorial Institue for the U.S. Department of Energy under contract DE-AC06-76RLO 1830. Contact David Feller, Karen Schuchardt, or Don Jones for further information.

[128] K. Fægri. NQvD basis sets. Technical report, Department of Chemistry, University of Oslo, 1994.

[129] D. B. Chesnut, B. E. Rusiloski, K. D. Moore, and D. A. Egolf. Use of locally dense basis sets for nuclear magnetic resonance shielding calculations. *J. Comput. Chem.*, 14:1364, 1993.

[130] F. B. van Duijneveldt. IBM Res. Rep. RJ945. Technical report, IBM, 1971.

[131] A.D. Becke. a. *Phys. Rev. A*, 38:3098, 1988.

[132] T.W. Keal and D.J. Tozer. The exchange-correlation potential in kohn-sham nuclear magnetic resonance shielding calculations. *jcp*, 119:3015, 2003.

[133] R. van Leeuwen and E.J. Baerends. Exchange-correlation potential with correct asymptotic behavior. *pra*, 1994.

[134] C. Lee, W. Yang, and R.G. Parr. a. *prb*, 57:785, 1988.

[135] B. Miehlich, A. Savin, H. Stoll, and H. Preuss. a. *cpl*, 157:200, 1989.

[136] N.C. Handy and A.J Cohen. a. *Mol. Phys.*, 99:403, 2001.

[137] J.P. Perdew. a. *prb*, 33:8822, 1986.

[138] J.P. Perdew and Y. Wang. a. *prb*, 33:8800, 1986.

[139] J.P. Perdew, J. A. Chevary, S.H. Vosko, K.A. Jackson, M.R. Pederson, and C. Fiolhais. a. *prb*, 46:6671, 1992.

[140] J.P. Perdew and A. Zunger. a. *prb*, 23:5048, 1981.

[141] P. Hohenberg and W. Kohn. a. *pr*, 136:B864, 1964.

[142] W. Kohn and L.J. Sham. a. *pr*, 140:A1133, 1965.

[143] S.H. Vosko, L. Wilk, and M. Nusair. a. *Can. J. Phys*, 58:1200, 1980.

[144] A.D. Becke. a. *jcp*, 98:5648, 1993.

[145] T. Yanai, D.P. Tw, and N.C. Handy. A new hybrid exchange-correlation functional using the Coulomb-attenuating method (cam-b3lyp). *cpl*, 393:51, 2004.

[146] T.W. Keal D.J. Tozer and T. Helgaker. Giao shielding constants and indirect spin-spin coupling constants: performance of density functional methods. *cpl*, 391:374, 2004.

[147] Keal and Tozer. A semi-empirical generalised gradient approximation exchange-correlation functional. *jcp*, 121:5654, 2004.

[148] C. Adamo and V. Barone. a. *jcp*, 110:6158, 1999.

[149] J.P. Perdew, K. Burke, and M. Ernzerhof. a. *prl*, 77(3865), 1996.

[150] C. W. Bauschlicher. The construction of modified virtual orbitals (MVO's) which are suited for configuration interaction calculations. *J. Chem. Phys.*, 72:880, 1980.

[151] T. U. Helgaker, H. J. Aa. Jensen, and P. Jørgensen. Analytical calcultion of MCSCF dipole-moment derivatives. *J. Chem. Phys.*, 182:6280, 1986.

[152] H. Hellmann. *Einführung in die Quantenchemie.* Deuticke, Leipzig, 1937.

[153] R. P. Feynman. *Phys. Rev.*, 56:340, 1939.

[154] V. Bakken, T. Helgaker, W. Klopper, and K. Ruud. The calculation of molecular geometrical properties in the Hellmann-Feynman approximation. *Mol. Phys.*, 96:653, 1999.

[155] O. Plashkevych, T. Privalov, H. Ågren, V. Carravetta, and K. Ruud. On the validity of the equivalent cores approximation for computing x-ray photoemission and photoabsorption spectral bands. *Chem. Phys.*, 260:11, 2000.

[156] P. Norman, K. Ruud, and T. Helgaker. *J. Chem. Phys.*, 120:5027, 2004.

[157] H. Sellers. *Int. J. Quantum Chem.*, 30:433, 1986.

[158] T. Helgaker. Translational and rotational symmetries of molecular geometrical derivatives. *Acta Chem. Scand.*, pages 515–518, 1988.

[159] J. Rose, T. Shibuya, and V. McKoy. *J. Chem. Phys.*, 58:74, 1973.

[160] T. Shibuya, J. Rose, and V. McKoy. *J. Chem. Phys.*, 58:500, 1973.

[161] E. R. Davidson. *J. Chem. Phys.*, 17:87, 1975.

[162] O. Vahtras, H. Ågren, and H. J. Aa. Jensen. Direct one-index transformations in multiconfiguration response calculations. *J. Comput. Chem.*, 15:573, 1994.

[163] J. Olsen, D. L. Yeager, and P. Jørgensen. Triplet excitation properties in large scale multiconfigurational linear response calculations. *J. Chem. Phys.*, 91:381, 1989.

[164] O. Vahtras, H. Ågren, P. Jørgensen, H. J. Aa. Jensen, T. Helgaker, and J. Olsen. Multiconfigurational quadratic response functions for singlet and triplet perturbations: the phosphoresence lifetime of formaldehyde. *J. Chem. Phys.*, 97:9178, 1992.

[165] P. Jørgensen, H. J. Aa. Jensen, and J. Olsen. Linear response for large scale multiconfigurational self-consistent field wave functions. *J. Chem. Phys.*, 89:3654, 1988.

[166] H. Hettema, H. J. Aa. Jensen, P. Jørgensen, and J. Olsen. Quadratic response functions for a multiconfigurational self-consistent field wave-function. *J. Chem. Phys.*, 97:1174, 1992.

[167] H. Ågren, O. Vahtras, H. Koch, P. Jørgensen, and T. Helgaker. Direct atomic orbital based self-consistent-field calculations of nonlinear molecular properties - application to the frequency-dependent hyperpolarizability of para-nitroaniline. *J. Chem. Phys.*, 98:6417, 1993.

[168] H. Ågren, O. Vahtras, and B. Minaev. Response theory and calculations of spin-orbit coupling phenomena in molecules. *Adv. Quantum Chem.*, 27:71, 1996.

[169] P. Norman, D. Jonsson, O. Vahtras, and H. Ågren. Cubic response functions in the random phase approximation. *Chem. Phys. Lett.*, 242:7, 1995.

[170] D. Jonsson, P. Norman, and H. Ågren. Cubic response functions in the multiconfigurational self-consistent field approximation. *J. Chem. Phys.*, 105:6401, 1996.

[171] D. Jonsson, P. Norman, Y. Liu, and H. Ågren. Response theory for static and dynamic polarizabilities of excited states. *J. Chem. Phys.*, 105:581, 1996.

[172] S. Koseki, M. W. Schmidt, and M. S. Gordon. *J. Phys. Chem.* **A**, 102:10430, 1998.

[173] O. Christiansen, H. Koch, and P. Jørgensen. The second-order approximate coupled cluster singles and doubles model CC2. *Chem. Phys. Lett.*, 243:409, 1995.

[174] C. Møller and M. S. Plesset. *Phys. Rev.*, 46:618, 1934.

[175] M. Head-Gordon, R. J. Rico, M. Oumi, and T. J. Lee. A doubles correction to electronic excited-states from configuration-interaction in the space of single substitutions. *Chem. Phys. Lett.*, 219:21, 1994.

[176] G. D. Purvis and R. J. Bartlett. *J. Chem. Phys.*, 76:1910, 1982.

[177] O. Christiansen, H. Koch, and P. Jørgensen. Perturbative triple excitation corrections to coupled cluster singles and doubles excitation energies. *J. Chem. Phys.*, 105:1451, 1996.

[178] K. Raghavachari, G. W. Trucks, J. A. Pople, and M. Head-Gordon. A 5th-order perturbation comparison of electron correlation theories. *Chem. Phys. Lett.*, 157:479, 1989.

[179] O. Christiansen, H. Koch, and P. Jørgensen. Response functions in the CC3 iterative triple excitation model. *J. Chem. Phys.*, 103:7429, 1995.

[180] H. Koch, O. Christiansen, P. Jørgensen, A. Sánchez de Merás, , and T. Helgaker. The CC3 model: An iterative coupled cluster approach including connected triples. *J. Chem. Phys.*, 106:1808, 1997.

[181] A. Halkier, H. Koch, O. Christiansen, P. Jørgensen, and T. Helgaker. First-order one-electron properties in the integral-direct coupled cluster singles and doubles model. *J. Chem. Phys.*, 107:849, 1997.

[182] A. Halkier and S. Coriani. Relaxed CC2 first-order properties. (Technical notes), 2000.

[183] K. Hald, A. Halkier, P. Jørgensen, S. Coriani, and T. Helgaker. A Lagrangian, integral-density direct formulation and implementation of the analytic CCSD and CCSD(T) gradients. *J. Chem. Phys.*, 118:2985, 2003.

[184] O. Christiansen, A. Halkier, H. Koch, P. Jørgensen, and T. Helgaker. Integral-direct coupled cluster calculations of frequency-dependent polarizabilities, transition probailities and excited-state properties. *J. Chem. Phys.*, 108:2801, 1998.

[185] O. Christiansen, P. Jørgensen, and C. Hättig. Response functions from Fourier component variational perturbation theory applied to a time-averaged quasienergy. *Int. J. Quantum Chem.*, 68:1, 1998.

[186] C. Hättig, O. Christiansen, and P. Jørgensen. Cauchy moments and dispersion coefficients using coupled cluster linear response theory. *J. Chem. Phys.*, 107:10592, 1997.

[187] O. Christiansen, H. Koch, A. Halkier, P. Jørgensen, T. Helgaker, and A. M. Sanchez de Meras. Large-scale calculations of excitation energies in coupled cluster theory: The singlet excited states of benzene. *J. Chem. Phys.*, 105:6921, 1996.

[188] C. Hättig K. Hald and P. Jørgensen. Triplet excitation energies in the coupled cluster singles and doubles model using an explicit triplet spin coupled excitation space. *J. Chem. Phys.*, 113:7765, 2000.

[189] J. Olsen K. Hald, C. Hättig and P. Jørgensen. Cc3 triplet excitation energies using an explicit excitation space. *J. Chem. Phys.*, 115:3545, 2001.

[190] S. Coriani, C. Hättig, P. Jørgensen, and T. Helgaker. Gauge-origin independent magneto-optical activity within coupled cluster response theory. *J. Chem. Phys.*, 113:3561, 2000.

[191] S. Coriani. *Ab initio determination of molecular properties*. PhD thesis, Aarhus University, Århus, Denmark, July 2000.

[192] C. Hättig and P. Jørgensen. Derivation of coupled cluster excited states response functions and multiphoton transition moments between two excited states as derivatives of variational functionals. *J. Chem. Phys.*, 109:4745, 1998.

[193] C. Hättig, O. Christiansen, S. Coriani, and P. Jørgensen. Frequency-dependent polarizabilities of excited states using coupled cluster response theory. *J. Chem. Phys.*, 109:9237, 1998.

[194] W. T. Raynes and R. Ratcliffe. *Mol. Phys.*, 37:571, 1979.

Part VI

Index

Index

346