

Dalton2018.0 – LSDALTON Program Manual

V. Bakken, A. Barnes, R. Bast, P. Baudin, S. Coriani, R. Di Remigio,
K. Dundas, P. Ettenhuber, J. J. Eriksen, L. Frediani, D. Friese, B. Gao,
T. Helgaker, S. Høst, I.-M. Høyvik, R. Izsák, B. Jansík, F. Jensen,
P. Jørgensen, J. Kauczor, T. Kjærgaard, A. Krapp, K. Kristensen, C. Kumar,
P. Merlot, C. Nygaard, J. Olsen, E. Rebolini, S. Reine, M. Ringholm,
K. Ruud, V. Rybkin, P. Sałek, A. Teale, E. Tellgren, A. Thorvaldsen,
L. Thøgersen, M. Watson, L. Wirz, and M. Ziolkowski

Contents

Preface	iv
I LSDALTON Installation Guide	1
1 Installation of LSDALTON	2
1.1 Installation instructions	2
1.2 Hardware/software supported	2
1.3 Source files	2
1.4 Installing the program using the Makefile	3
1.5 The Makefile.config	5
1.5.1 Intel compiler	5
1.5.2 Gfortran/gcc compiler	9
1.5.3 Using 64-bit integers	12
1.5.4 Using Compressed-Sparse Row matrices	13
II LSDALTON User's Guide	14
2 Getting started with LSDALTON	15
2.1 The MOLECULE input file	16
2.1.1 BASIS	17
2.1.2 ATOMBASIS	18
2.1.3 User defined basis sets	19
2.2 The LSDALTON.INP file	21
3 Interfacing to LSDALTON	24
3.1 The Grand Canonical Basis	24
3.2 Basisset and Ordering	24
3.3 Normalization	25

3.4	Matrix File Format	26
3.5	The LSDALTON library bundle	27
III LSDALTON Reference Manual		29
4	List of LSDALTON keywords	30
4.1	**GENERAL	31
4.1.1	*TENSOR	32
4.2	**INTEGRAL basic keywords	32
4.3	**INTEGRAL advanced keywords	35
4.3.1	*FMM	36
4.4	**WAVE FUNCTION	37
4.4.1	*DFT INPUT	41
4.4.2	*DENSOPT	44
4.4.3	\$INFO	50
4.5	**OPTIMIZE	50
4.6	**DYNAMI	54
4.7	**LOCALIZE ORBITALS	55
4.8	**RESPONS	58
4.8.1	*SOLVER	59
4.8.2	*DIPOLE	60
4.8.3	*DIPOLEMOMENTMATRIX	60
4.8.4	*ALPHA	61
4.8.5	*BETA	62
4.8.6	*DAMPED_TPA	62
4.8.7	*QUASIMCD	63
4.8.8	*SHIELD	65
4.8.9	*INASHIELD	66
4.8.10	*ESGRAD	66
4.8.11	*GAMMA	67
4.8.12	*MOLGRA	68
4.8.13	*TPA	68
4.8.14	*NTO	68
4.9	**OPENRSP	69
4.9.1	Disclaimer	69
4.9.2	Citation guide	69
4.9.3	Introduction	70
4.9.4	General settings	71

4.9.5	*SOLVER	72
4.9.6	*RESPONSE FUNCTION	72
4.9.7	*RESPONSE FUNCTION example	74
4.9.8	Format of returned data from OPENRSP	76
4.9.9	Other optional OPENRSP input	78
4.9.10	OPENRSP test cases	80
4.10	**DEC and **CC	81
4.10.1	**DEC input keywords	81
4.10.2	**CC input keywords	85
4.10.3	Common **DEC and **CC input keywords	87
4.11	**PLT	89
4.12	**PLTGRID	91
4.13	**PCM	92
4.14	**WANNIER	92

Preface

This is the manual for the LSDALTON quantum chemistry program — Release Dalton2018.0 — for computing Hartree-Fock and DFT wave functions, energies, and molecular properties. For correlated models, MP2 geometry optimizations and CCSD energies (not linear-scaling) are available. Most parts of LSDALTON employ *linear scaling* and massively parallel implementations, which makes it suitable for calculations on large molecular systems, in particular when the calculations are carried out on large super computer architectures.

If results obtained with this code are published the following paper should be cited:

K. Aidas, C. Angeli, K. L. Bak, V. Bakken, R. Bast, L. Boman, O. Christiansen, R. Cimiraglia, S. Coriani, P. Dahle, E. K. Dalskov, U. Ekström, T. Enevoldsen, J. J. Eriksen, P. Etenhuber, B. Fernández, L. Ferrighi, H. Fliegl, L. Frediani, K. Hald, A. Halkier, C. Hättig, H. Heiberg, T. Helgaker, A. C. Hennum, H. Hettema, E. Hjertenæs, S. Høst, I.-M. Høyvik, M. F. Iozzi, B. Jansik, H. J. Aa. Jensen, D. Jonsson, P. Jørgensen, J. Kauczor, S. Kirpekar, T. Kjærgaard, W. Klopper, S. Knecht, R. Kobayashi, H. Koch, J. Kongsted, A. Krapp, K. Kristensen, A. Ligabue, O. B. Lutnæs, J. I. Melo, K. V. Mikkelsen, R. H. Myhre, C. Neiss, C. B. Nielsen, P. Norman, J. Olsen, J. M. H. Olsen, A. Osted, M. J. Packer, F. Pawłowski, T. B. Pedersen, P. F. Provasi, S. Reine, Z. Rinkevicius, T. A. Ruden, K. Ruud, V. Rybkin, P. Salek, C. C. M. Samson, A. Sánchez de Merás, T. Saue, S. P. A. Sauer, B. Schimelpfennig, K. Snegov, A. H. Steindal, K. O. Sylvester-Hvid, P. R. Taylor, A. M. Teale, E. I. Tellgren, D. P. Tew, A. J. Thorvaldsen, L. Thøgersen, O. Vahtras, M. A. Watson, D. J. D. Wilson, M. Ziolkowski, and H. Ågren, "*The Dalton quantum chemistry program system*", **WIREs Comput. Mol. Sci.** (doi: 10.1002/wcms.1172)

We emphasize the conditions under which the program is distributed. It is furnished for your own use, and you may not redistribute it further, neither in whole nor in part. Even though LSDALTON is completely separate from the original DALTON program, the program packages are distributed together in the Dalton2018.0 suite. Thus, anyone interested in obtaining LSDALTON should check out the DALTON homepage at <http://daltonprogram.org>. Visit <http://dalton-installation.readthedocs.org> for

comprehensive installation instructions, and visit the Forum <http://daltonprogram.org/forum> for tutorials on how to run LSDALTON and other useful information. The Forum is also open for general discussions, and is the place to look/ask for assistance if you do not find the answers you are looking for in this manual.

Finally note that 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.

Part I

LSDALTON **Installation Guide**

Chapter 1

Installation of LSDALTON

1.1 Installation instructions

LSDALTON is configured using CMake, typically via the setup script, and subsequently compiled using make or gmake.

Please consult <http://dalton-installation.readthedocs.org> for details. Help and support from the Dalton community is available at <http://daltonprogram.org/forum>.

1.2 Hardware/software supported

LSDALTON can be run on a variety of systems running the UNIX operating system. The current release of the program supports Linux, Cray, SGI, and MacOS using GNU, Intel, PGI and XL compilers.

The program is written in FORTRAN 90 and C, with machine dependencies isolated using C preprocessor directives. All floating-point computations are performed in 64-bit precision.

The program should be portable to other UNIX platforms. 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.

1.3 Source files

The Dalton2018.0 program suite is distributed as a tar file obtainable from the Dalton homepage at <http://www.daltonprogram.org>. After you have downloaded the file extract it with:

```
tar xvzf DALTON-2015.*-Source.tar.gz
```


Most of the extracted subdirectories under `LSDALTON/*` contain source code for the different sections constituting the LSDALTON program. Furthermore, there is a directory containing various public domain routines (`pdpack`), a 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_LSDALTON`).

In addition to the above directories, the main dalton directory contain several files that support the CMake build mechanism (`CMakeLists.txt` and `cmake/*`) as well as a directory containing all the basis sets supplied with this distribution (`basis`).

1.4 Installing the program using the Makefile

The LSDALTON program can, in addition to the standard standard CMake method (see section 1.1), be built using a distributed Makefile and configure script.

The main `.../LSDALTON` directory contains a shell script (`configure`). Based on the automatic detection of the architecture, 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
```

The script will now try to detect the architecture, which usually works fine for most common platforms. However, if for some reason this is not possible, you may choose one of the supported architectures from a list (currently comprising only three architectures: `aix`, `linux`, and `darwin`):

Although this script is in most cases capable of making a correct `Makefile.config`, we always recommend users to check the created `Makefile.config` against local system set-up. In particular, check that `Makefile.config` contains the path(s) to the proper mathematical libraries. The program runs many times slower if not linked to any mathematical libraries (i.e. using LSDALTON's own "hand coded" matrix multiplications etc.)

During the execution of the `configure` script, you will be asked to decide which compiler you would like to use.

The configure script searches for compilers in a ordered fashion, and in each instance you can choose to accept the compiler. When a suitable compiler has been found, you will be asked a few questions:

1. Do you want to use debug compiler options ? [y/N]

Specifies whether or not you would like to use low optimization level and debug options, useful for developments. The y and N in the square bracket indicate the two options. y=Yes, N=No. The Capital letter is the default. In this case the default (chosen by pressing enter) is No.

2. Do you want to install the program with OpenMP parallelization? [Y/n]

Specifies whether or not you would like to compile the code using openMP (utilizing several cores on a single node).

Note that the configure script do not support MPI parallelization and the resulting Makefile.config have to be manually changed to use MPI.

Default is Y=Yes.

3. Does the Fortran compiler support the Fortran 2003 features like procedure pointers and iso_c_bindings?**4. Use current directory as installation directory for binaries and scripts?**

Denotes the directory where the executable and the run script will be moved to.

5. Use basis as default basis set directory?

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. Changes to the basis set should be done in a separate directory, and you may then supply this basis set directory to the program at execution time using the command.

```
> export BASDIR=path/to/your/basisetdir
```

In order to ensure that your basis set is read correctly we recommend to use the

```
**INTEGRALS  
.BASPRINT  
6
```

Which will print the exponents and contraction coefficients read from file as well as the normalized basis used in LSDALTON.

6. Default scratch space

Determines the default head scratch directory where temporary files will be placed. This value will be put in the `lsdalton` 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.

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

```
> make
```

1.5 The Makefile.config

In this section we give a brief introduction to the `Makefile.config`, which contains compiler instructions for the compilation of the program. The introduction is directed to new users with a limited understanding of Makefiles and we only discuss subjects that may be relevant. Note that the configure script will in most cases provided a `Makefile.config` which works and no modifications are required from the user. Depending on the version of math libraries, the configure script may not be able to find these and it may be required to manually add paths to mathematical libraries.

1.5.1 Intel compiler

The first lines of a `Makefile.config` using the intel ifort/icc compilers may look like this

```
1  ARCH          = linux
2  FMMDIR        = mm
3  #
4  #
5  CPPFLAGS      = -DSYS_LINUX -D_FILE_OFFSET_BITS=64
   -D'INSTALL_BASDIR="/home/user/lsdalton/basis/"' -DVAR_LINSCA
   -DVAR_OMP -DCOMPILER_UNDERSTANDS_FORTRAN_2003 -DVAR_IFORT
6  F77           = ifort
7  F90           = ifort
8  FLNK          = ifort
9  CC            = icc
10 CXX           = icpc
11 RM            = rm -f
12 FFLAGS        = -O3 -ip -w
13 F90OPTFLAGS   = -O3 -ip -w -fpp1 -openmp
14 SAFEFFLAGS    = -O2 -w
15 CFLAGS        = -O3 -ip -restrict -DRESTRICT=restrict -openmp -DUSE_UNDERSCORES
16 CXXFLAGS      = -Wall -g -wd981 -wd279 -wd383 -vec-report0 -wd1572 -wd177 -fno-rtti
   -fno-exceptions -O3 -openmp -DUSE_UNDERSCORES
```

```

17 INCLUDES      =
18 LIBS          = -lstdc++ -openmp -liomp5 -lpthread
19 INSTALLEDIR   = /home/user/lsdalton
20 PDPACK_EXTRAS = linpack.o eispack.o cholesky.o invroutines.o
gp_dlapack.o gp_zlapack.o gp_dblas3.o gp_dblas2.o gp_dblas1.o gp_zblas.o

```

We now describe this Makefile.config file line by line, before we discuss a number of issues.

- Line 1: ARCH is set to the architecture. (for an aix architecture ARCH=rs6000, for a Mac computer ARCH=darwin, these will be discussed shortly)
- Line 2: The directory of the Fast multipole moment code
- Line 5: CPPFLAGS is a list of precompiler flags which is required for the program to compile correctly

-DSYS_LINUX tells the LSDALTON program that this is a linux architecture

-DINSTALL_BASDIR The basisset installation directory

-DVAR_LINSCA Is a mandatory keyword!

-DVAR_OMP tells the LSDALTON program that OpenMP is used

-DCOMPILER_UNDERSTANDS_FORTRAN_2003 tells the LSDALTON program that the Fortran compiler support the Fortran 2003 features like procedure pointers and iso_c_bindings

-DVAR_IFORT tells the LSDALTON program that the Fortran compiler is an Intel ifort compiler

Usually this line works out of the box

- Line 6: specifies the compiler used to compile fortran 77 files (obsolete)
- Line 7: specifies the compiler used to compile fortran 90 files
- Line 8: specifies the compiler used for linking
- Line 9: specifies the compiler used to compile c files
- Line 10: specifies the compiler used to compile c files
- Line 11: specifies the command to be used when cleaning
- Line 12-16: specifies compiler flags that should be used to compile fortran files. Naturally these flags depend on the compiler. Here we use a intel fortran compiler (ifort)

- O3 Specifies the code optimization for applications. Possible options are O0, O1, O2, or O3, with O2 as default
 - ip Additional interprocedural optimizations for single-file compilation are enabled.
 - w Disables all warning messages (same as -nowarn)
 - fpp1 Runs the Fortran preprocessor on source files before compilation. Syntax is fpp[n], where n can be 0, 1, 2, or 3, where 1, 2, or 3 tells the compiler to run the preprocessor
 - openmp Enables the parallelizer to generate multi-threaded code based on the OpenMP directives. This option sets option automatic, which causes all local, non-saved variables to be allocated to the run-time stack, this may result in a lack of stack-memory as discussed below.
- Line 15: specifies compiler flags that should be used to compile C files. Naturally these flags depend on the compiler. Here we use a intel fortran compiler (icc). Some of the compiler options have already been explained under the ifort options and they will not be explained again.
 - Line 16: specifies compiler flags that should be used to compile C++ files.
 - restrict Pointer disambiguation is enabled with the restrict qualifier.
 - DRESTRICT=restrict Precompiler flag used in DFT code
 - DUSE_UNDERSCORES required to define the communication between fortran and C files.
 - Line 17: Empty by default. Used if for some reason, any additional directories should be included.
 - Line 18: Libraries you wish to include. -openmp (used to be -lguide) and -lpthread are required for openmp parallelization. See Sections [1.5.1.2](#) and [1.5.2.1](#) for more details on how to link to mkl and other libraries. If you use the old version of ifort you need to replace -openmp with -lguide.
 - Line 19: the install directory
 - Line 20: LSDALTON provides its own blas and lapack libraries, located in the pdpack directory. These should only be used if no mathematical libraries are available, since they slow down the code significantly.

In case you link to an external library, like the intel mkl library, you only need to include "linpack.o eispack.o cholesky.o invroutines.o".

In case you do not link to an external library, you need to include "linpack.o eispack.o cholesky.o invroutines.o gp_dlapack.o gp_zlapack.o gp_dblas3.o gp_dblas2.o gp_dblas1.o gp_zblas.o."

1.5.1.1 OpenMP

Note that when compiling using the `-openmp` option it may be required to increase the stack memory. On a linux machine the size of the stack-memory can be viewed using the command

```
> ulimit -a
```

and changed by

```
> ulimit -s newstacksize
```

The newstacksize should be a large number or unlimited.

1.5.1.2 Math Library: MKL

Linking to a math library is crucial for optimal performance. The linking depends on the library available. We encourage to first compile with the default Makefile.config composed by the configure script. This ensures that your chosen compiler works with the LSDALTON source code, and then you can try to link to a math library. There is a number of different ways that can be used to link LSDALTON with the mkl library. An example of a brute force way is the following:

```
16 LIBS          =
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_solver_lp64_sequential.a
-Wl,--start-group /opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_intel_lp64.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_sequential.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_core.a -Wl,--end-group
-lguide -lpthread
17 INSTALLDIR    = /home/user/lsdalton
18 PDPACK_EXTRAS = linpack.o eispack.o
```

for the sequential linking (no OpenMP), assuming that the intel compiler is located at `/opt/intel/Compiler/11.1/056` and that the mkl library is located at `/opt/intel/Compiler/11.1/056`.

This link line have been obtained from the "Intel Math Kernel Library Link Line Advisor" at <http://software.intel.com/en-us/articles/intel-mkl-link-line-advisor>.

The OpenMP version of the library can be used in the following way.

```

16 LIBS          =
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_solver_lp64_sequential.a
-Wl,--start-group /opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_intel_lp64.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_intel_thread.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_core.a -Wl,--end-group
-lguide -lpthread
17 INSTALLDIR    = /home/user/lsdalton
18 PDPACK_EXTRAS = linpack.o eispack.o

```

A somewhat more elegant way is to add the MKL directory to the INCLUDE line

```

15 INCLUDES      = -I/home/user/lsdalton/include
16 LIBS          = -I/opt/intel/Compiler/11.1/056/mkl/include -lmkl_intel_lp64
-lmkl_sequential -lmkl_core
17 INSTALLDIR    = /home/user/lsdalton
18 PDPACK_EXTRAS = linpack.o eispack.o

```

or a threaded version

```

15 INCLUDES      = -I/home/user/lsdalton/include
16 LIBS          = -I/opt/intel/Compiler/11.1/056/mkl/include -lmkl_intel_lp64
-lmkl_intel_thread -lmkl_core -lguide -lpthread
17 INSTALLDIR    = /home/user/lsdalton
18 PDPACK_EXTRAS = linpack.o eispack.o

```

1.5.2 Gfortran/gcc compiler

The first lines of a Makefile.config using the gfortran/gcc (version 4.5.1) on a linux architecture may look like this

```

1 ARCH          = linux
2 FMMDIR        = mm
3 #
4 #
5 CPPFLAGS      = -DSYS_LINUX -D_FILE_OFFSET_BITS=64
-D'INSTALL_BASDIR="/home/user/lsdalton/basis/' -DVAR_LINSCA
-DVAR_OMP -DCOMPILER_UNDERSTANDS_FORTRAN_2003 -DGFORTRAN=472
6 F77           = gfortran
7 F90           = gfortran
8 FLNK          = gfortran
9 CC            = gcc

```

```

10 CXX          = g++
11 RM           = rm -f
12 FFLAGS       = -march=x86-64 -O3 -ffast-math -funroll-loops
-ftree-vectorize -ffloat-store
13 F90OPTFLAGS  = -march=x86-64 -O3 -ffast-math -funroll-loops
-ftree-vectorize -ffloat-store -I. -x f95-cpp-input -ffloat-store -fopenmp
14 SAFEFFLAGS   = -march=x86-64 -O3 -ffast-math -funroll-loops
-ftree-vectorize -ffloat-store
15 CFLAGS       = -march=x86-64 -O3 -ffast-math -funroll-loops
-ftree-vectorize -std=c99 -DRESTRICT=restrict -DFUNDERSCORE=1 -ffloat-store
-DUSE_UNDERSCORES
16 CXXFLAGS     = -march=x86-64 -g -O3 -Wall -fno-rtti -fno-exceptions -fopenmp -DUSE_UNDERSCORES
17 INCLUDES     = -I/home/user/lsdalton/include
18 LIBS         = -lstdc++ -lgomp
19 INSTALLDIR   = /home/user/lsdalton
20 PDPACK_EXTRAS = linpack.o eispack.o cholesky.o invroutines.o gp_dlapack.o
gp_zlapack.o gp_dblas3.o gp_dblas2.o gp_dblas1.o gp_zblas.o

```

On a MAC computer a Makefile.config using gfortran/gcc could look like this

```

1 ARCH         = darwin
2 FMMDIR       = mm
3 #
4 #
5 CPPFLAGS     = -DSYS_LINUX -D_FILE_OFFSET_BITS=64
-DVAR_SPLITFILES -DVAR_G77 -DGFORTTRAN=472 -DVAR_LINSCA
-D'INSTALL_BASDIR="/home/user/lsdalton/basis/'
-DVAR_OMP -DCOMPILER_UNDERSTANDS_FORTRAN_2003
6 F77          = gfortran
7 F90          = gfortran
8 FLNK        = gfortran
9 CC           = gcc
10 CXX         = g++
11 RM          = rm -f
12 FFLAGS      = -O3 -ffast-math -fexpensive-optimizations -funroll-loops
-ftree-vectorize
13 F90OPTFLAGS = -O3 -ffast-math -fexpensive-optimizations -funroll-loops
-ftree-vectorize -I. -x f95-cpp-input
14 SAFEFFLAGS  = -O2 -ffast-math -fexpensive-optimizations -funroll-loops
-ftree-vectorize

```



```

15 CFLAGS          = -O3 -ffast-math -fexpensive-optimizations -funroll-loops
   -ftree-vectorize -std=c99 -DRESTRICT=restrict -DFUNDERSCORE=1 -DUSE_UNDESCORES
16 CXXFLAGS       = -O3 -Wall -fno-rtti -fno-exceptions -fopenmp -DUSE_UNDESCORES
17 INCLUDES      = -I/home/user/lldalton/include
18 LIBS          = -lstdc++ -lpthread -lm -lgfortranbegin -lgfortran
19 INSTALDIR     = /home/user/lldalton
20 PDPACK_EXTRAS = linpack.o eispack.o cholesky.o invroutines.o gp_dlapack.o
   gp_zlapack.o gp_dblas3.o gp_dblas2.o gp_dblas1.o gp_zblas.o

```

We now describe these Makefile.config files in comparison to section [1.5.1](#).

Line 1 to 5 is unchanged for the linux architecture, but on the MAC machine - DGFORTTRAN must be included in the CPPFLAGS.

Line 6-9 specifies the use of the gfortran/gcc compilers.

Line 11-13 specify compiler flags and are naturally different for different compilers

`-march=x86-64` This specifies the name of the target architecture. GCC uses this name to determine what kind of instructions it can emit when generating assembly code. In this case it is a 64 bit machine. The use of this compiler flag is not required, but recommended.

`-O3` Specifies the code optimization for applications. Possible options are O0, O1, O2, or O3, with O2 as default

`-ffast-math` A Compiler option to optimize floating-point arithmetic

`-funroll-loops` Unroll loops whose number of iterations can be determined at compile time or upon entry to the loop

`-ftree-vectorize` Perform loop vectorization on trees. This flag is enabled by default at `-O3`

`fexpensive-optimizations` Perform a number of minor optimizations that are relatively expensive.

`-ffloat-store` This may be required depending on the gfortran version, but usually not. it enforces IEEE compliance

`-fopenmp` Enables the parallelizer to generate multi-threaded code based on the OpenMP directives.

`-std=c99` Determines the language standard, in this case C 99 standard.

Line 16 are libraries you wish to include. `-lgomp` are required for openmp parallelization.

1.5.2.1 Math Library: Linux MKL

Linking to a math library is crucial for optimal performance. The linking depends on the library available. We encourage to first compile with the default Makefile.config composed by the configure script. This ensures that your chosen compiler works with the LSDALTON source code, and then you can try to link to a math library. LSDALTON provides its own blas and lapack libraries, which is placed in the pdpack directory. However it is encouraged to link to an external library, like the intel mkl library, then you only need to include linpack.o eispack.o. In the case that you do not link to an external library, you need to include all. linpack.o eispack.o gp_dlapack.o gp_zlapack.o gp_dblas3.o gp_dblas2.o gp_dblas1.o gp_zblas.o.

Note that linking to mkl library using OpenMP requires to use -liomp5 in stead of -lgomp

```
17 INCLUDES      = -I/home/user/lsdalton/include
-I/opt/intel/Compiler/11.1/056/mkl/include
18 LIBS          = -lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core
-lguide -lpthread -liomp5
19 INSTALLDIR    = /home/user/lsdalton
20 PDPACK_EXTRAS = linpack.o eispack.o
```

or

```
17 INCLUDES      =
18 LIBS          =
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_solver_lp64_sequential.a
-Wl,--start-group /opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_intel_lp64.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_intel_thread.a
/opt/intel/Compiler/11.1/056/mkl/lib/em64t/libmkl_core.a -Wl,--end-group
-lguide -lpthread -liomp5
19 INSTALLDIR    = /home/user/lsdalton
20 PDPACK_EXTRAS = linpack.o eispack.o
```

1.5.3 Using 64-bit integers

For large scale calculations, it may be necessary to compile the program using 64-bit integers instead of the default 32-bit integers. To do this, you need to include "-DVAR_INT64" in the CPPFLAGS (in addition to what is already there). Furthermore, the FFLAGS, F90OPTFLAGS, SAFEFFLAGS, and CFLAGS must be changed depending on you compiler. For ifort/icc, you need to add "-i8", and for XLF/XLC, you need to add "-qintsize=8". The use of 64-bit integers has not been tested for gfortran. Note that you also have to change

the mathematical libraries in LIBS to their 64-bit counterparts. This is not always a simple task, and you may need to ask your system administrator.

1.5.4 Using Compressed-Sparse Row matrices

When using Compressed-Sparse Row (CSR) matrices, only non-zero elements in matrices are stored. When matrices are sparse enough, linear scaling in both memory and cpu time is obtained. However, for small or non-sparse systems, there is a considerable overhead in cpu time compared to using standard dense matrices, so don't use it unless you need it. You enable CSR by putting "-DVAR_MKL" and "-DVAR_CSR" in your CPPFLAGS (in addition to what is already there. You also need to link to MKL's CSR library (what we have in LSDALTON is an interface to MKL CSR). Furthermore, to run a calculation using CSR matrices, you always need .CSR under *DENSOPT in the input file LSDALTON.INP (described in detail in chapters [2](#) and [4](#)).

Part II

LSDALTON User's Guide

Chapter 2

Getting started with LSDALTON

In this chapter we give an introduction to the two input files needed for doing a calculation with the LSDALTON program (MOLECULE.INP and LSDALTON.INP), 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.

Note! The DALTON and LSDALTON programs use different input keywords! Some MOLECULE.INP files used for LSDALTON can also be used for DALTON – but NOT vice versa! The DALTON.INP input file of DALTON is different from the LSDALTON.INP input file of LSDALTON. Thus, DALTON keywords CANNOT be assumed to work in LSDALTON or vice versa.

Before going into detail with the MOLECULE.INP and LSDALTON.INP files in chapters 2.1 and 2.2, we provide a simple input example for the impatient reader. The example below will yield the HF energy for water and the two lowest lying excitation energies and their associated transition strengths:

MOLECULE.INP:

```
BASIS
cc-pVDZ Aux=Ahlrichs-Coulomb-Fit
water R(OH) = 0.95Aa , <HOH = 109 deg.
Distance in Aangstroms
Atomtypes=2 Angstrom
Charge=8.0 Atoms=1
0      0.00000  0.00000  0.00000
Charge=1.0 Atoms=2
```

```
H      0.55168   0.77340   0.00000
H      0.55168  -0.77340   0.00000
```

LSDALTON.INP:

```
**WAVE FUNCTIONS
.HF
**RESPONS
.NEXCIT
2
*END OF INPUT
```

To run this simple example, you may (i) create a new directory, (ii) copy the two files above into this directory, and (iii) run the calculation from your newly created directory using the `lsdalton.x` executable located in your build directory (as mentioned above, installation details are given at <http://dalton-installation.readthedocs.org>).

The converged HF energy can be found in the output file LSDALTON.OUT by grepping for "Final HF energy":

```
Final HF energy:                -76.026476337669
```

while the two excitation energies and associated transition strengths are given at the end of LSDALTON.OUT:

```
*****
*                               ONE-PHOTON ABSORPTION RESULTS (in a.u.)                               *
*****
Excitation      Transition Dipole Moments      Oscillator
Energies        x          y          z          Strengths
=====
0.33897330      -0.94893963E-15  -0.30390933E-15  0.36349099      0.29858056E-01
0.42634048      0.57831041      -0.15070410E-14  -0.14539976E-14  0.95057706E-01
```

Note that if you run this test your results may vary slightly from the above due to numerical inaccuracies.

2.1 The MOLECULE input file

Basically, the MOLECULE file contains the following information:

- The Cartesian coordinates and nuclear charges of all atoms
- The basis set(s) to be used

Please note the following:

- Supplying the molecular geometry in the form of a Z-matrix is not supported by LSDALTON since this format is ill suited for large scale calculations.
- Molecular symmetry is not exploited by LSDALTON (since the target systems of the program are large biomolecules, which hardly ever contains any symmetry).
- The LSDALTON program does not support the use of effective core potentials.

There are two different types of molecule files, using either BASIS or ATOMBASIS. ATOMBASIS is relevant only if different basis sets on individual atoms are required. In the first two subsections, we describe these two types of MOLECULE files using simple examples. The basis can be selected from the provided basis-set library or can be specified by the user (as described under section 2.1.3).

2.1.1 BASIS

Here we describe our example water molecule input file line by line using the BASIS MOLECULE format:

```

1  BASIS
2  cc-pVDZ Aux=Ahlrichs-Coulomb-Fit
3  water R(OH) = 0.95Aa , <HOH = 109 deg.
4  Distance in Aangstroms
5  Atomtypes=2 Angstrom
6  Charge=8.0 Atoms=1
7  0      0.00000  0.00000  0.00000
8  Charge=1.0 Atoms=2
9  H      0.55168  0.77340  0.00000
10 H      0.55168 -0.77340  0.00000

```

- Line 1: The word "BASIS", indicating that the same basis set should be used for all atoms.
- Line 2: The name of the basis set, in this case cc-pVDZ. Optional: The name of the auxiliary basis set, in this case Ahlrichs-Coulomb-Fit (only referenced if density-fitting is requested in LSDALTON.INP)

- Lines 3-4: Comments (may be left blank)
- Line 5: **Atomtypes=** Number of different atoms - in the case of H₂O, there are two different atom types, H and O. **Angstrom** If this keyword is omitted, the program will assume that coordinates are given in atomic units (1bohr=0.5291772083Å). Other options in this line are: **Charge=** Molecular charge - assumed to be zero if omitted. **Nosymmetry** This have no effect as no pointgroup symmetry have been implemented. **SubSystems** Use Subsystem labels for doing Counter Poise Corrected interaction energies
- Lines 6 and 8: A line of this type has to come before the Cartesian coordinates of each atom type, indicating a) **Charge=** the nuclear charge of the atom type, and b) **Atoms=** the number of atoms of this type. Other options in this line are: **SubSystem=A** Use Subsystem label A for these atoms **phantom** Place Phantom atom (add Basis functions but no actual atom) **pointcharge** place pointcharge without Basis functions
- Lines 7, 9-10: Cartesian coordinates of the atoms (in Angstrom, since this was specified in line 5). An atom name (1-4 characters long) must be placed at the beginning of these lines.

2.1.2 ATOMBASIS

Our example water molecule may also be specified as an ATOMBASIS MOLECULE file :

```

1  ATOMBASIS
2  water R(OH) = 0.95Aa , <HOH = 109 deg.
3  Distance in Aangstroms
4  Atomtypes=2 Angstrom
5  Charge=8.0 Atoms=1 Basis=cc-pVDZ Aux=Ahlrichs-Coulomb-Fit
6  O      0.00000  0.00000  0.00000
7  Charge=1.0 Atoms=2 Basis=cc-pVDZ Aux=Ahlrichs-Coulomb-Fit
8  H      0.55168  0.77340  0.00000
9  H      0.55168 -0.77340  0.00000

```

As seen from the example, there is little difference between the two formats. Line 2 in the BASIS file has been removed, and instead the basis sets (and optionally, the auxiliary basis sets) are now given for each atom. In this example they are the same, as it would probably not make much sense to use different basis sets for the atoms in water. This can make sense, though, for larger molecules. An example could be a biomolecule containing an active site with one or more transition metal atoms. It may be useful to describe such an active site with a more sophisticated basis set than the surroundings.

2.1.3 User defined basis sets

The user may supply his or her own basis set in two ways. Both ways require that the MOLECULE.INP file is argumented with a basis as in the following two examples

```

1  BASIS
2  USERDEFINED
3  water R(OH) = 0.95Aa , <HOH = 109 deg.
4  Distance in Aangstroms
5  Atomtypes=2 Angstrom
6  Charge=8.0 Atoms=1
7  O      0.00000  0.00000  0.00000
8  Charge=1.0 Atoms=2
9  H      0.55168  0.77340  0.00000
10 H      0.55168 -0.77340  0.00000
11
12 USERDEFINED BASIS
13 a 1
14 $ HYDROGEN      (4s,1p) -> [2s,1p]
15 $ HYDROGEN      segmented contracted
16 $ S-TYPE FUNCTIONS
17      4      2      0
18      18.7311370  0.03349460  0.00000000
19      2.8253937  0.23472695  0.00000000
20      0.6401217  0.81375733  0.00000000
21      0.1612778  0.00000000  1.00000000
22 $ P-TYPE FUNCTIONS
23      1      1      0
24      1.1000000  1.00000000
25 a 8
26 $ OXYGEN        (10s,4p) -> [3s,2p]
27 $ S-TYPE FUNCTIONS
28      10      3      0
29      5484.6717000  0.00183110  0.00000000  0.00000000
30      825.2349500  0.01395010  0.00000000  0.00000000
31      188.0469600  0.06844510  0.00000000  0.00000000
32      52.9645000  0.23271430  0.00000000  0.00000000
33      16.8975700  0.47019300  0.00000000  0.00000000
34      5.7996353  0.35852090  0.00000000  0.00000000

```

```

35      15.5396160  0.00000000 -0.11077750  0.00000000
36      3.5999336  0.00000000 -0.14802630  0.00000000
37      1.0137618  0.00000000  1.13076700  0.00000000
38      0.2700058  0.00000000  0.00000000  1.00000000
39  $ P-TYPE FUNCTIONS
40      4      2      0
41      15.5396160  0.07087430  0.00000000
42      3.5999336  0.33975280  0.00000000
43      1.0137618  0.72715860  0.00000000
44      0.2700058  0.00000000  1.00000000
45  $ D-TYPE FUNCTIONS
46      1      1      0
47      0.8000000  1.00000000

```

Line 13 to 24 contain the Basis set information for the Hydrogen atom. Line 13 denote the nuclear charge. Line 14 through 16 are comments. Line 17 states that there are 4 primitive functions and 2 contracted functions. Lines 18 to 21 contain information about the primitive functions. The first number is the exponent and the next 2 numbers are the coefficients for the 2 contracted functions. Line 22-24 contain info about p orbitals on hydrogen, while Line 25-47 contain similar information about oxygen.

Alternative the userdefined basis may be used in connection with the ATOMBASIS

ATOMBASIS

water R(OH) = 0.95Aa , <HOH = 109 deg.

Distance in Aangstroms

Atomtypes=2 Angstrom

Charge=8.0 Atoms=1 Basis=cc-pVDZ

O 0.00000 0.00000 0.00000

Charge=1.0 Atoms=2 Basis=USERDEFINED

H 0.55168 0.77340 0.00000

H 0.55168 -0.77340 0.00000

USERDEFINED BASIS

a 1

\$ HYDROGEN (4s) -> [2s]

\$ HYDROGEN (1p)

\$ S-TYPE FUNCTIONS

4 2 0

18.7311370 0.03349460 0.00000000

```
2.8253937 0.23472695 0.00000000
0.6401217 0.81375733 0.00000000
0.1612778 0.00000000 1.00000000
$ P-TYPE FUNCTIONS
1 1 0
1.1000000 1.00000000
```

Finally the basis set files from the Basis Set exchange (EMSL) <https://bse.pnl.gov/bse/portal> may be downloaded and put into the basis set directory (the directory called "basis" located at the root of the Dalton library) and used directly. The name specified by the input should match the name of the file in the basis set directory (see testcase LSDALTON/test/linsca/linsca_emsl for an example).

2.2 The LSDALTON.INP file

In this section, we show two typical examples of a LSDALTON.INP file, one for small molecules and one for large molecules. Many other examples of input files for requesting e.g. specific molecular properties may be found at <http://daltonprogram.org>, and a complete reference manual of all keywords is found in Chapter 4.

In general, the LSDALTON.INP is divided in different sections under the headlines (see Section 4 for details):

- ****GENERAL** contains general settings (optional)
- ****INTEGRAL** contains settings for the calculation of integrals (optional)
- ****WAVE FUNCTION** contains information about the wave function (e.g. HF/DFT) and settings for the optimization of the wave function (mandatory)
- ****OPTIMIZE** contains settings for geometry optimization (optional)
- ****DYNAMI** contains settings for Born-Oppenheimer molecular dynamics (optional)
- ****LOCALIZE ORBITALS** contains settings for orbital localization procedure (optional)
- ****RESPONS** contains information about requested molecular properties for HF and DFT (optional)
- ****DEC** or ****CC** contains info about MP2 and coupled-cluster calculations (optional)
- ****PLT** contains information about construction of *.plt files which may be used to visualize densities and orbitals using e.g. the Chimera program [1] (optional).

- ****PLTGRID** contains information about grid used for construction of *.plt files (optional).
- ****PCM** contains information about inclusion of a continuum description of the solvent *via* the polarizable continuum model [2, 3].

Each of these sections may contain subsections, indicated by a single asterisk. The LSDALTON.INP file should always end with ***END OF INPUT**.

A typical example of LSDALTON.INP for a **small molecule** could look like:

```
**WAVE FUNCTIONS
.HF
*DENSOPT
.RH
.DIIS
.CONVTHR
1.0D-6
**RESPONS
.NEXCIT
5
*END OF INPUT
```

Here, we have requested a Hartree-Fock optimization under ****WAVE FUNCTION**. The subsection ***DENSOPT** contains setting for how the density should be optimized. The **.RH** (i.e. Roothaan-Hall) and **.DIIS** keywords request a standard diagonalization combined with the DIIS scheme for convergence acceleration. With the keyword **.CONVTHR** (i.e. convergence threshold) it is requested that iterations are terminated when the Frobenius norm of the SCF gradient, divided by the square root of the number of occupied orbitals, is smaller than 10^{-6} . Finally, we have under ****RESPONS** requested the calculation of the five lowest excitation energies with **.NEXCIT** (i.e. number of excitation energies).

A typical example of LSDALTON.INP for a **large molecule** could look like:

```
**INTEGRAL
.DENSFIT
**WAVE FUNCTIONS
.DFT
BP86
*DENSOPT
.START
TRILEVEL
.ARH
```

```
.CONVDYN  
STANDARD  
*END OF INPUT
```

Under ****INTEGRAL**, we have requested the use of density-fitting to speed up the calculation (the use of J-engine is default). Under ****WAVE FUNCTIONS**, we have requested a DFT calculation using the BP86 exchange-correlation functional.

Since this is a calculation on a large molecule, we request the trilevel starting guess (atomic, valence, and then full optimization) and the Augmented Rothaan-Hall (ARH) method for density optimization. Note that ARH is default, so this keyword is actually unnecessary. ARH is more robust than the standard diagonalization/DIIS scheme. Furthermore, the ARH scaling with system size is asymptotically linear provided sparse-matrix algebra is employed. You may use sparse-matrix algebra by putting **.CSR** (i.e. Compressed-Sparse Row) under ***DENSOPT** (NB: requires linking to the MKL library). We have used a dynamic SCF convergence threshold (**.COVNDYN**, see 4.4.2). This is suitable for calculations on large molecules, since the standard SCF convergence threshold is based on the Frobenius norm of the SCF gradient, which is not size-extensive. The use of the **STANDARD** dynamic threshold corresponds to 10^{-5} times the square root of the number of electrons in the system.

Chapter 3

Interfacing to LSDALTON

In this chapter we give an introduction to the normalization, basis set and ordering of atomic orbitals used in the LSDALTON program.

This is provided in case you want to interface to the LSDALTON program or want to read one of the files written by the LSDALTON program, for instance the local orbitals. Certain functionality can also be incorporated into other programs by linking to the LSDALTON-library bundle, as outlined in section 3.5.

3.1 The Grand Canonical Basis

Unlike all other Quantum Chemistry Programs the LSDALTON program, uses the so called grand canonical basis [4, 5] as the internal default basis. The results therefore cannot directly be compared to other programs unless the

`.NOGCBASIS`

keyword is specified. This keyword deactivates the use of the grand canonical basis and uses the input basis instead. An exception is Dunning’s correlation consistent basis sets (cc-pVDZ, cc-pVTZ, ..., aug-cc-pVDZ, etc.) for which the `.NOGCBASIS` keyword is used by default.

3.2 Basisset and Ordering

In LSDALTON the default basis set are real-valued spherical harmonic Gaussian type orbitals (GTOs)

$$G_{ilm}(\mathbf{r}; a_i; \mathbf{A}) = S_{lm}(x_A; y_A; z_A) \exp(-a_i r_A^2) \quad (3.1)$$

where $S_{lm}(x_A; y_A; z_A)$ is a real solid harmonic introduced in section 6.4.2 of Ref. [6] (see Table 6.3). Concerning the ordering of basis functions, it is one atomtype after the other

as given in the input, and for each atomtype, it is one atom after another as given in input. For each atom increasing with angular momentum l , within one angular momentum l , the ordering is first the components (e.g. p_x, p_y, p_z), and then the contracted functions.

So for an atom consisting of 3 s functions 2 p functions and 1 d function, the ordering is $\{1s; 2s; 3s; 1p_x; 1p_y; 1p_z; 2p_x; 2p_y; 2p_z; 1d_{xy}; 1d_{yz}; 1d_{z^2-r^2}; 1d_{xz}; 1d_{x^2-y^2}\}$ In LSDALTON the ordering of the basis functions are $m = \{-l, \dots, 0, \dots, l\}$ for a given angular momentum l . Although for computational efficiency the p orbitals are treated as a special case with $p = \{p_x, p_y, p_z\}$ which corresponds to $m = \{1, -1, 0\}$.

3.3 Normalization

For an unnormalized primitive spherical harmonic GTO, the overlap is given by

$$\langle \chi_{ilm}^{\text{GTO}} | \chi_{ilm}^{\text{GTO}} \rangle = (N_i^{\text{P}})^2 \left(\frac{\pi}{2a_i} \right)^{3/2} \frac{1}{(4a_i)^l} \quad (3.2)$$

(where the last term corresponds to $1/(2p)^l$ according to for example Eq. 9.3.8 of Ref. [6]. This gives the primitive normalization factor N^{P}

$$N_i^{\text{P}} = \frac{(4a_i)^{l/2+3/4}}{(2\pi)^{3/4}} \quad (3.3)$$

Naturally the default in LSDALTON is not primitive GTOs but contracted GTOs written as linear combinations of primitive spherical-harmonic GTOs of different exponents

$$G_{\mu lm}(\mathbf{r}; a; \mathbf{A}) = \sum_i G_{ilm}(\mathbf{r}; a_i; \mathbf{A}) d_{i\mu}^{\text{norm}} \quad (3.4)$$

in LSDALTON we determine $d_{i\mu}^{\text{norm}}$ as

$$d_{i\mu}^{\text{norm}} = N_i^{\text{P}} N_i^{\text{C}} d_{i\mu}^{\text{basis}} \quad (3.5)$$

As explained, N_i^{P} is the normalization of the primitive GTOs, $d_{i\mu}^{\text{basis}}$ are the unmodified contraction coefficients read from the basis set file. N_i^{C} is the normalization of the contracted functions and is determined using the overlap between two normalized primitive GTOs of the same angular momentum, but different exponents

$$\langle \chi_{a_i lm}^{\text{GTO}} | \chi_{a_j lm}^{\text{GTO}} \rangle = \left(\frac{\sqrt{4a_i a_j}}{a_i + a_j} \right)^{\frac{3}{2}+l} \quad (3.6)$$

so that the contraction coefficients for the contracted function N_i^{C} are found by first finding the overlap

$$\mathcal{S}_{\mu} = \sum_{ij} d_{i\mu}^{\text{basis}} d_{j\mu}^{\text{basis}} \langle \chi_{ilm}^{\text{GTO}} | \chi_{jlm}^{\text{GTO}} \rangle = \sum_{ij} d_{i\mu}^{\text{basis}} d_{j\mu}^{\text{basis}} \left(\frac{\sqrt{4a_i a_j}}{a_i + a_j} \right)^{\frac{3}{2}+l} \quad (3.7)$$

where $d_{i\mu}^{\text{basis}}$ are the unmodified coefficients and then construct the coefficients

$$d_{i\mu}^{\text{norm}} = N_i^{\text{p}} N_{i\mu}^{\text{c}} d_{i\mu}^{\text{basis}} = N_i^{\text{p}} \frac{d_{i\mu}^{\text{basis}}}{\sqrt{S_{\mu}}} = \frac{d_{i\mu}^{\text{basis}}}{\sqrt{S_{\mu}}} (4a_i)^{\frac{1}{2} + \frac{3}{4}} \left(\frac{1}{2\pi} \right)^{\frac{3}{4}} \quad (3.8)$$

3.4 Matrix File Format

During an LSDALTON calculation a number of files may be generated. This list of files include

dens.restart - The density matrix
 fock.restart - The fock matrix
 overlapmatrix - The overlap matrix
 cmo_orbitals.u - The canonical Molecular orbitals
 lcm_orbitals.u - The local Molecular orbitals

the files are all written using the Fortran 90 code

```
1 OPEN(UNIT=IUNIT, FILE='dens.restart', STATUS='UNKNOWN', FORM='UNFORMATTED', IOSTAT=
  IOS)
2 WRITE(iunit) A%Nrow, A%Ncol
3 WRITE(iunit) (A%elms(I), I=1, A%nrow*A%ncol)
4 CLOSE(UNIT=IUNIT, STATUS='KEEP')
```

Where A is a derived type containing the number of rows (A%nrow), the number of columns (A%ncol), and the elements stored in a vector array (A%elms).

The dens.restart is different as it contains an additional logical

```
1 OPEN(UNIT=IUNIT, FILE='dens.restart', STATUS='UNKNOWN', FORM='UNFORMATTED', IOSTAT=
  IOS)
2 WRITE(iunit) A%Nrow, A%Ncol
3 WRITE(iunit) (A%elms(I), I=1, A%nrow*A%ncol)
4 WRITE(iunit) GCBASIS
5 CLOSE(UNIT=IUNIT, STATUS='KEEP')
```

IMPORTANT: When interfacing to other programs, the logical GCBASIS should always be false, i.e. by specifying the keyword

.NOGCBASIS

in the LSDALTON.INP. If the logical GCBASIS is true the dens.restart is given in terms of the grand canonical basis rather than the input basis.

3.5 The LSDALTON library bundle

The LSDALTON library bundle `liblsdalton.a` allows for porting LSDALTON code functionality to other programs. To build `liblsdalton.a` add `-DENABLE_LSLIB=ON` to the `setup` command. Currently this functionality is limited to the evaluation of certain standard AO integrals and integral components: the overlap matrix, the Cartesian multipole moment integrals, the one-electron integral matrix (combining kinetic energy and nuclear-electron attraction contributions), the Coulomb, exchange and exchange-correlation matrices and the four-center two-electron repulsion integrals. Also, the first- to third-order derivatives of these integral components with respect to the nuclear displacements are available, see the file `LSDALTON/lsdaltonsrc/LSlib_tester.F90` for details. Note that the mixed electrical and geometrical derivatives can be obtained from the dipole moment geometrical derivatives.

After building LSDALTON under a `build` directory (consult the installation guide <http://dalton-installation.readthedocs.org> for assistance), the LSDALTON library `liblsdalton.a` can be found under `build\lib`. You can now link your program to the LSDALTON library. The use is perhaps best illustrated by a simple example. The file `lslib_test.F90`

```
PROGRAM testlslib
implicit none
integer :: nbast,natoms,nelectrons,lupri,luerr
integer :: i,j
double precision, allocatable :: Smat(:,,:)

call lsinit_all()

lupri = 6 !logical unit number of standard output
luerr = 0 !logical unit number of standard error
CALL LSlib_get_dimensions(nbast,natoms,nelectrons,lupri,luerr)
allocate(Smat(nbast,nbast))
CALL LSlib_get_overlap(Smat,nbast,lupri,luerr)

deallocate(Smat)
call lsfree_all()

END PROGRAM testlslib
```

calculates the AO overlap matrix (`Smat`). To compile `lslib_test.F90` you have to link to the external libraries required by LSDALTON, typically `lapack` and `blas`. For an example `intel/mkl` build

```
./setup --fc=ifort --cc=icc --cxx=icpc --mkl=parallel --omp -DENABLE_LSLIB=ON
```

you can build the executable `lslib_test.x` from `lslib_test.F90` according to

```
ifort -mkl=parallel -openmp -parallel -O3 lslib_test.F90 -o lslib_test.x  
/dalton-path/build/lib/liblsdalton.a
```

Exactly how to compile and link your code to `liblsdalton.a` for other types of build varies, and it can then be instructive to see how `LSDALTON` is buildt. To see how `lsdalton.x` is compiled and linked, you can type

```
make VERBOSE=1 lsdalton.x
```

in the build directory. The actual build of `lsdalton.x` can be found towards the end of the standard output.

The execution of `lslib_test.x` requires the precense of valid `MOLECULE` and `LS-DALTON` input files in the directory where you execute your program, for example the `MOLECULE.INP`

```
BASIS  
STO-3G
```

```
Atomtypes=2  
Charge=8.0 Atoms=1  
O 0.00000 0.00000 0.00000  
Charge=1.0 Atoms=2  
H 0.00000 0.50000 0.00000  
H 0.00000 0.00000 0.50000
```

and the `LSDALTON.INP`

```
**GENERAL  
.NOGCBASIS  
**WAVE FUNCTION  
.DFT  
BLYP  
*END OF INPUT
```

Part III

LSDALTON Reference Manual

Chapter 4

List of LSDALTON keywords

In this chapter, we describe all keywords for the different sections of the LSDALTON.INP file. In general, LSDALTON.INP is divided into the following sections under the headlines:

- ****GENERAL** contains general settings (optional)
- ****INTEGRAL** contains settings for the calculation of integrals (optional)
- ****WAVE FUNCTION** contains information about the wave function (e.g. HF/DFT) and settings for the optimization of the wave function (mandatory)
- ****OPTIMIZE** contains settings for geometry optimization (optional)
- ****DYNAMI** contains settings for Born-Oppenheimer molecular dynamics (optional)
- ****LOCALIZE ORBITALS** contains settings for orbital localization procedure (optional)
- ****RESPONS** contains information about requested molecular properties for HF and DFT (optional)
- ****DEC or **CC** contains info about MP2 and coupled-cluster calculations (optional)
- ****PLT** contains information about construction of *.plt files which may be used to visualize densities and orbitals using e.g. the Chimera program [1] (optional).
- ****PLTGRID** contains information about grid used for construction of *.plt files (optional).
- ****PCM** contains information needed to set up a polarizable continuum model calculation using the PCMSolver external module [7] (optional).

Each of these sections may contain subsections, indicated by a single asterisk. LSDALTON.INP should always end with *END OF INPUT.

4.1 **GENERAL

This input module contains general settings.

- .INTERACTIONENERGY Calculation of the Counter Poise Corrected Interaction energy (Hartree-Fock, DFT, DEC-MP2 and CCSD).
NB: Requires the use of SubSystem labels in the MOLECULE.INP file
- .SAMESUBSYSTEMS Assumes that the two subsystems of the Counter Poise Corrected Interaction energy calculation is identical.
- .SUBSYSTEMDENSITY Constructs a density matrix for the subsystems of the Counter Poise Corrected Interaction energy calculation using the full system density matrix.
- .SCALAPACK Use SCALAPACK matrices.
NB: Requires linking to the scalapack lib (provided by MKL) and the use of the `-scalapack` keyword during installation. This keyword will distribute the memory of matrices among the MPI processes. This keyword will also use the SCALAPACK library to paralyze the matrix operations like matrix multiplication.
- .SCALAPACKBLOCKSIZE
<ScalapackBlockSize>
The blocksize used in SCALAPACK matrices (not required)
- .SCALAPACKNODES
<ScalapackNodes>
The number of MPI processes you wish to use for the SCALAPACK format. This must be a subset of the total number of MPI processes used. It may be beneficial to chose a subset of the total MPI processes because the matrix operations scale differently with MPI nodes than the Integral evaluation.
- .CSR Use Compressed-Sparse Row matrices. NB: Requires linking to MKL library and the use of the `-csr` keyword during installation!
- .TIME Activate the printout of timings.
- .NOGCBASIS Deactivate the use of the Grand Canonical basis [4, 5]
- .FORCEGCBASIS The Grand Canonical basis [4, 5] is deactivated when the program detects the use of a Dunning basis set like (cc-pVXZ) but the calculation will force the use of the Grand Canonical basis using this keyword.
- .PSFUN Use the older DFT functional library of Salek, instead of the XCFun library. See the documentation of .DFT under **WAVE FUNCTION for more information.

4.1.1 ***TENSOR**

This input submodule of ****GENERAL** contains settings specific for setting the behaviour of the underlying tensor library ScaTeLib.

.DEBUG Set debugging information and safe execution variables.

.SEGMENT_LENGTH

`<SegmentLength>`

Set the max segment length for tiled distributed tensors in each of the tensor dimensions.

4.2 ****INTEGRAL** basic keywords

This input module defines the way the integrals should be calculated.

.ADMM Use the Auxiliary Density Matrix Method (ADMM) [8, 9] to approximate the exact-exchange contribution; by projecting the density to a smaller basis for calculation of the exact exchange, and by including a GGA exchange correction between the regular and smaller auxiliary basis. When using this keyword, an auxiliary basis set has to be specified in the MOLECULE file such as:

```
1    BASIS
2    6-31+G* Aux=df-def2 ADMM=3-21G
    ...
```

where the "ADMM" basis is used as an auxiliary basis set for ADMM (and where "Aux" specify the auxiliary basis for density fitting (optional)). The **.ADMM** keyword defaults to ADMM2 with Becke type exchange for the GGA correction term (identical to using **.ADMM2** together with **.ADMM-FUNC** specified to B88X). Note that although calculations requesting ADMM and reponse properties are allowed, the response part of the code does not actually apply the ADMM approximation. ADMM energy and gradients are implemented.

.ADMM1 Use the ADMM1 approximation of Ref. [8]. Note: This approximation does not work for gradients!

.ADMM2 Use the ADMM2 approximation of Ref. [8]. Default.

.ADMMS Use the ADMMS approximation of Ref. [9]. Reccomended.

- .ADMMP Use the ADMMP approximation of Ref. [9].
- .ADMMQ Use the ADMMQ approximation of Ref. [9]. Not recommended.
- .ADMM-FUNC Reads a GGA exchange functional from the next input line. Specifies the GGA exchange functional to use for the correction term. List of possible functionals: B88X, PBEX, RPBEX, REVPBEX, MPBEX, PW91X, KT1X, KT2X, KT3X, G96X, LG93X and OPTX. KT3X together with ADMMS works well for basis-set combinations 6-31G**/3-21G and cc-pVTZ/3-21G [9].
- LDAX is also implemented, but is currently not working as expected. Its use is therefore discouraged.
- .AOPRINT
<Print level>
Print the Atomic Orbital information. The higher the print level, the more information will be printed. Default value is 0.
- .BASPRINT
<Print level>
Print the basis set information. The higher the print level, the more information will be printed. Using a print level of 6 will print the input exponents and contraction coefficients read from file as well as the normalized basis used in LSDALTON.
- .DENSFIT Use density-fitting for the Coulomb contribution (no exact exchange density-fitting method have been introduced). When using this keyword, an auxiliary basis set has to be specified in the MOLECULE file. For the Poples basis sets we recommend df-def2 and for Dunning's basis sets use the corresponding cc-pVXZdenfit basis sets, with $X = T, Q, 5$.
- .LINSCAPRINT
<Print level>
Print level in the integral code. The higher the print level, the more information will be printed. This keyword corresponds to
- .AOPRINT
<Print level>
.BASPRINT
<Print level>
.MOLPRINT
<Print level>

See the individual keywords for more details. Default value is 0.

.MOLPRINT

<Print level>

Print information about the molecule. The higher the print level, the more information will be printed. Default value is 0.

.NOJENGINE Turn off J-engine algorithm (which is default) for the calculation of the Coulomb matrix (see Refs. [10] and [11]). Compute instead Coulomb-like contributions from the explicitly calculated integrals.

.NOLINK Turn off the LinK algorithm (which is default) for the calculation of the exchange matrix (see Ref. [12]).

.NO SCREEN Deactivate the use of all screening methods.

.NOFAMILY Deactivate the exploitation of family type basis set, basis set sharing exponents for different angular momentums.

.PARI Use the Pair-Atomic Resolution-of-the-Identity approximation [13] for the Coulomb and exact Exchange contribution. When using this keyword, an auxiliary basis set has to be specified in the MOLECULE file.

We do not recommend to use this method for any production runs, and have included it in case other method developers would like to explore it (see the reference for details).

.RUNMM Use the (Fast) Multipole Method.

.THRESH

<Threshold>

An overall screening threshold for integral evaluation. The default value is 10^{-8}

The various integral-evaluation thresholds (below) are set according to this Threshold.

- The Cauchy-Schwarz screening threshold is set equal to Threshold (can be set separately by **.THR_CS**)
 - The Screening threshold used for Coulomb is $\text{Threshold} \cdot 10^{-2}$ (Default: 10^{-10})
 - The Screening threshold used for Exchange is $\text{Threshold} \cdot 10^{-0}$ (Default: 10^{-8})
 - The Screening threshold used for One-electron operators is $\text{Threshold} \cdot 10^{-7}$ (Default: 10^{-15})
- The Primitive Cauchy-Schwarz screening threshold is set equal to $\text{Threshold} \cdot 10^{-1}$ (can be set separately by **.THR_PS**)

- The Overlap-distribution distance-screening threshold is set equal to Threshold $\cdot 10^{-1}$. When calculating overlap integrals, this threshold is used for setting up AO extents. Overlap distributions (ODs) for which the distance between the two AOs are larger than the sum of the extents are screened away.
- The Overlap-distribution extent-screening threshold is set equal to Threshold $\cdot 10^{-1}$. When calculating overlap integrals, this threshold is used for setting up OD extents. Overlap integrals between two ODs separated by more than the sum of the extents are screened away.
- The FMM threshold is used to distinguish the Coulomb repulsion into classical and non-classical interactions, based on the non-classical extent of the continuous charge distribution (orbitals or orbital products). By default this threshold is set equal to Threshold $\cdot 10^{-1}$ (can be set separately by `.SCREEN` under `*FMM`)

4.3 ****INTEGRAL** advanced keywords

This input module defines the way the integrals should be calculated. This section contains some advanced keywords which is mostly useful for debugging or comparisons with other codes.

`.NO CS` Deactivate the use of Cauchy-Schwarz screening. Note that this does not deactivate the Primitive Cauchy-Schwarz screening.

`.NO PS` Deactivate the use of primitive Cauchy-Schwarz screening.

`.THR_CS`

`<CS_Threshold>`

Cauchy-Schwarz screening threshold. The default value is 10^{-8} . Note that

- The Screening threshold used for Coulomb is `CS_Threshold` $\cdot 10^{-2}$ (Default: 10^{-10})
- The Screening threshold used for Exchange is `CS_Threshold` $\cdot 10^{-0}$ (Default: 10^{-8})
- The Screening threshold used for One-electron operators is `CS_Threshold` $\cdot 10^{-7}$ (Default: 10^{-15})

`.THR_PS`

`<PS_Threshold>`

Primitive Cauchy-Schwarz screening threshold. The default value is 10^{-9}

- .CART-E Use cartesian E-coefficients instead of hermite E-coefficients for the McMurchie-Davidson integral-evaluation scheme [14] used in LSDALTON. The use of Cartesian E-coefficients is default in most integral programs. We however use hermite E-coefficients according to ref. [15].

- .FTUVMAXPRIM
 <Maxprim>
Sets the maximum number of primitives used in the FTUV batches in the J-engine algorithm.

- .LOW RJ000 ACCURACY Use a decreased accuracy for the calculation of the Boys function.

- .MAXPASSES
 <maxpass>
Change the maximum number of collected overlap distributions (default is 40).

- .NO PASS Deactivate the use of Passes, the collection of overlap distributions which is used as default in order to increase efficiency.

- .NOSEGMENT Deactivate the use of segments. The use of segments is used to reduced the number of primitive functions in the calculation of integrals. The basis is considered to be a general contracted basis where all primitives contribute to all contracted functions.

- .NSETUV Use non-spherical E-coefficients.

- .OVERLAP-DF-J Use the Overlap density fitting algorithm for the calculation of the Coulomb matrix.

- .UNCONT Treat the basis functions (given by input) as fully uncontracted basis functions (i.e. ignore any contraction coefficients given in the basis and treat instead all the primitives as separate basis functions).

4.3.1 *FMM

This input block is used to specify settings for the (continuous) fast multipole moment (FMM) treatment of classical Coulomb interactions.

For gradients OpenMP parallelization is turned off. The reason for this is that a continuous counter used to identify moments for orbital products has to be the same in both moments and derivative moments. For the same reason screening [only during writing (see printmm routines)!!!] is turned off both for the serial and parallel cases. As a consequence of the screening been turned off the moments have to be recalculated for the gradient and can not

be reused from the preceding energy evaluation. The gradient implementation has been tested both for regular and density fitting Coulomb interactions. It works for the combined N-e + e-e + N-N interactions (default), and for the e-e interaction only (invoked by `.NOONE`). It has however not been tested for speed and efficiency.

`.LMAX`

<Lmax>

The maximum multipole-moment expansion order of a given orbital product. Default Value is 8.

`.TLMAX`

<TLmax>

The maximum order used for translated multipole-moments. Default Value is 20.

`.SCREEN`

<Threshold>

The FMM threshold is used to distinguish the Coulomb repulsion into classical and non-classical interactions, based on the non-classical extent of the continuous charge distribution (orbitals or orbital products). By default this threshold is set equal to $\cdot 10^{-9}$. Defines the threshold used to determine if a contribution can be calculated classically or non-classically.

`.NOONE` Do not use FMM for the nuclear-electron attraction.

`.NOMMBU` Do not use io-buffers for interfacing multipole-moments to the FMM driver.

4.4 **WAVE FUNCTION

`.HF` Hartree-Fock calculation.

`.DFT`

<FUNCTIONAL>

DFT calculation.

The exchange—correlation functionals in LSDALTON can be divided into two groups: generic and combined functionals. A number of generic functionals are included within LSDALTON. The combined functionals are linear combinations of generic ones.

It should be noted that the input is not case sensitive, although the notation employed in this manual makes use of case to emphasise exchange or correlation functional properties and to reflect the original literature sources.

4.4.0.1 Exchange Functionals

Slater Dirac-Slater exchange functional [16, 17, 18].

Becke 1988 Becke exchange GGA correction [19]. Note that the full Becke88 exchange functional is given as Slater + Becke.

OPTX Handy's 2001 exchange functional correction [20]. The full OPTX exchange functional is given by $1.05151*\text{Slater} - 1.43169*\text{OPTX}$.

PBEx Perdew, Burke and Ernzerhof 1996 exchange functional [21].

PW86x Perdew and Wang 1986 exchange functional (the PWGGA-I functional)[22].

4.4.0.2 Correlation Functionals

VWN3 correlation functional of Vosko, Wilk and Nusair, 1980 (equation III) [23]. This is the form used in the Gaussian program.

VWN5 correlation functional of Vosko, Wilk and Nusair, 1980 (equation V – the recommended one). The VWN keyword is a synonym for VWN5 [23].

LYP correlation functional by Lee, Yang and Parr, 1988 [24, 25].

P86c non-local part of the correlation functional of the Perdew 1986 correlation functional [26]. PZ81 (1981 Perdew local) is usually used for the local part of the functional, with a total correlation functional of $\text{P86c} + \text{PZ81}$.

PBEc Perdew, Burke and Ernzerhof 1996 correlation functional, defined as PW91c local and PBEc non-local correlation [21].

PW91c 1991 correlation functional of Perdew and Wang (the pwGGA-II functional) [27]. This functional includes both the PW91c non-local and PW91c local (ie PW92c) contributions. The non-local PW91c contribution may be determined as $\text{PW91c} - \text{PW92c}$.

PZ81 local correlation functional of Perdew and Zunger, 1981 [28].

4.4.0.3 Standalone Functionals

LB94 asymptotically correct functional of Leeuwen and Baerends 1994 [29]. This functional improves description of the asymptotic density on the expense of core and inner valence.

4.4.0.4 Combined functionals

GGAKey is a universal keyword allowing users to manually construct arbitrary linear combinations of exchange and correlation functionals from the list above.

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 . . .`. As an example, B3LYP may be constructed as:

```
.DFT
GGAKey EXX=0.2 Slater=0.8 Becke=0.72 LYP=0.81 VWN=0.19
```

The following GGA and hybrid functional aliases are defined within LSDALTON and provide further examples of the GGAKey keyword.

SVWN5 is a sum of Slater functional and VWN (or VWN5) correlation functional. It is equivalent to

```
GGAKey Slater=1 VWN5=1
```

SVWN3 is a sum of the Slater exchange functional and VWN3 correlation functional.

It is equivalent to the Gaussian program LSDA functional and can alternatively be selected by following set of keywords

```
GGAKey Slater=1 VWN3=1
```

LDA A synonym for SVWN5.

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 3-parameter hybrid functional [30] equivalent to:

```
GGAKey EXX=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. Keyword B3LYPGauss is a synonym for B3LYPg. This functional can be explicitly set up by

```
GGAKey EXX=0.2 Slater=0.8 Becke=0.72 LYP=0.81 VWN3=0.19
```

BP86 Becke88 exchange functional and Perdew86 correlation functional (with Perdew81 local correlation). The explicit form is:

```
GGAKey Slater=1 Becke=1 PZ81=1 P86c=1
```

B3P86 variant of B3LYP with VWN used for local correlation and P86 for the nonlocal part.

```
GGAKey EXX=0.2 Slater=0.8 Becke=0.72 P86c=0.81 VWN=1
```

B3P86-G variant of B3LYP with VWN3 used for local correlation and P86 for the non-local part. This is the form used by the Gaussian quantum chemistry program.

```
GGAKey EXX=0.2 Slater=0.8 Becke=0.72 P86c=0.81 VWN3=1
```

BPW91 Becke88 exchange functional and PW91 correlation functional. The explicit form is:

```
GGAKey Slater=1 Becke=1 PW91c=1
```

CAMB3LYP Coulomb Attenuated Method Functional of Yanai, Tew and Handy [31].

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 exchange correction [32, 33].

```
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 (not equal to 1.0) [32, 33].

```
GGAKey Slater=1.07173 VWN=0.576727 KT=-0.006
```

KT3 a hybrid functional of Slater, OPTX and KT exchange with the LYP correlation functional [34]. 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 [20, 24, 25].

```
GGAKey Slater=1.05151 OPTX=-1.43169 LYP=1
```

PBE0 a hybrid functional of Perdew, Burke and Ernzerhof with 0.25 weight of exact exchange, 0.75 of PBE exchange functional and the PBE correlation functional [35]. Alternative aliases are PBE1PBE or PBE0PBE.

```
GGAKey EXX=0.25 PBE=0.75 PBE=1
```

PBE same as above but with exchange estimated exclusively by PBE exchange functional [21]. Alias of PBE1PBE. This is the form used by CADPAC and NWChem quantum chemistry programs.

```
GGAKey PBE=1 PBE=1
```

Note that the Molpro quantum chemistry program uses the PW91c non-local correlation functional instead of PBE, which is equivalent to the following:

```
GGAKey PBE=1 PW91c=1 .
```

Note that combinations of local and non-local correlation functionals can also be generated with the GGAKey keyword. For example, `GGAKey P86c=1 PZ81=1` combines the PZ81 local and P86c non-local correlation functional, whereas `GGAKey VWN=1 P86c=1` combines the VWN local and P86 non-local correlation functionals.

Linear combinations of all exchange and correlation functionals listed above are possible with the GGAKey keyword.

4.4.0.5 Functional implementation details

Starting with LSDALTON1.0 the functionals are evaluated using the XCFun library[36] by default, unless this is disabled at build time. The functional library of Pawel Salek *et al*[37] is still available and can be activated using the .PSFUN keyword under **GENERAL. For open-shell calculations the use of XCFun is strongly encouraged, but for closed shell calculations the .PSFUN option may result in faster calculations. Because of the small differences between the two different implementations it may not be possible to obtain exactly the same results using XCFun or .PSFUN, but the difference should be very small (i.e. energy differences of about 10^{-8}).

4.4.1 *DFT INPUT

Controls the XC-integration grid. The grid is generated as follows: For each atom a radial grid is created (different radial grids available, see below), which is multiplied with an angular grid (Lebedev-grids). The angular grids are by default pruned for radial points close to the nucleus, following Murray, Handy and Laming. [38] Grid weights for the grid points are calculated following different schemes (see below). Grid points with weights below 10^{-20} are disregarded. The default settings corresponds to .GRID4, but with the BLOCKSSF rather than the BLOCK partitioning scheme (see details below). Please note that the grid construction routines in the LSDALTON program changed compared to the DALTON program which may result in small deviations in the number of grid points etc.

.ANGINT

<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 31. Maximum value is 64. Note that the value of ANGINT is changed by the following keywords: COARSE, NORMAL, FINE, ULTRAFINE. The GRIDX keywords also imply specific ANGINT values.

.COARSE Shortcut keyword for radial integration accuracy 10^{-11} and angular expansion order equal to 35.

.DFTELS

<DFTELS>

safety threshold – stop if the charge integration error becomes larger then this threshold.

.DFTTHR

<DFTHRO, DFTHRL, DFTHRI, RHOTHR>

DFTHRO is not used

DFTHRL is not used

DFTHRI threshold for screening product of gaussian atomic orbitals

RHOTHr threshold for screening of the electron density

.DISPER defaults to .DFT-D3BJ (see below)

.DFT-D2 switches on Grimmes DFT-D2 empirical dispersion correction [39]. The code will attempt to assign the correct functional dependent parameters based on the chosen DFT functional. Analytic gradient contributions are available.

.D2PAR READ (LUINP,*) D2_s6_inp, D2_alp_inp, D2_rs6_inp
using this keyword user input values of the s_6 , α and $s_{r,6}$ DFT-D2 parameters may be specified. If supplied these values override any values defined within the code.

.DFT-D3 switches on Grimmes DFT-D3 empirical dispersion correction [40]. The code will attempt to assign the correct functional dependent parameters based on the chosen DFT functional. Analytic gradient contributions are available.

.DFT-D3BJ switches on Grimmes DFT-D3 empirical dispersion correction with Becke-Johnson damping [41]. The code will attempt to assign the correct functional dependent parameters based on the chosen DFT functional. Analytic gradient contributions are available. This is the presently recommended version.

.3BODY keyword for adding 3-body terms to the DFT-D3 dispersion energy. Note that gradients are not implemented for these corrections

.D3PAR READ (LUINP,*) D3_s6_inp, D3_alp_inp, D3_rs6_inp, D3_rs18_inp, D3_s18_inp
keyword for specifying the s_6 , α , $s_{r,6}$, $s_{r,8}$ and s_8 parameters of the DFT-D3 methods. Note, take care to match the parameter values to the correct version of the DFT-D3 correction.

.FINE Shortcut keyword for radial integration accuracy 10^{-13} and angular expansion order equal to 42.

.GRID TYPE

<GC2 LMG TURBO BECKE BECKEORIG SSF BLOCK BLOCKSSF>

GC2, LMG, TURBO define radial quadrature schemes, only one can be set. BECKE, BECKEORIG, SSF, BLOCK, BLOCKSSF define the grid partitioning schemes, only one can be set.

GC2 Gauss-Chebyshev quadrature of second kind.

- LMG As proposed by Lindh, Malmqvist and Gagliardi (default). [42]
- TURBO Treutler-Ahlrichs M4-T2 scheme. [43] Implies also that the angular integration quality becomes Z-dependant (see also ANGINT) and that pruning is used (see also NOPRUN).
- BECKE Becke partitioning scheme with atomic size correction. [44]
- BECKEORIG Becke partitioning scheme without atomic size correction (default). [44]
- SSF Stratmann-Scuseria-Frisch partitioning scheme. [45]
- BLOCK Becke partitioning scheme with atomic size correction [44] combined with a blockwise handling of grid points [46]. Useful for large molecules.
- BLOCKSSF Stratmann-Scuseria-Frisch partitioning scheme [45] combined with a blockwise handling of grid points [46]. Useful for large molecules.
- .GRID1 Shortcut for radial integration accuracy $1.0 * 10^{-5}$, angular expansion order equal to 17, radial quadrature TURBO, grid partitioning scheme BLOCK.
- .GRID2 Shortcut for radial integration accuracy $2.15447 * 10^{-7}$, angular expansion order equal to 23, radial quadrature TURBO, grid partitioning scheme BLOCK.
- .GRID3 Shortcut for radial integration accuracy $4.64159 * 10^{-9}$, angular expansion order equal to 29, radial quadrature TURBO, grid partitioning scheme BLOCK.
- .GRID4 Shortcut for radial integration accuracy $5.01187 * 10^{-14}$, angular expansion order equal to 35, radial quadrature TURBO, grid partitioning scheme BLOCK.
- .GRID5 Shortcut for radial integration accuracy $2.15443 * 10^{-17}$, angular expansion order equal to 47, radial quadrature TURBO, grid partitioning scheme BLOCK.
- .HARDNESS
<HARDNESS>
sets the hardness of the partitioning function in the Becke weighting scheme. Only positive integer values allowed. The higher the hardness the more stepfunction like the partitioning functions. Default value is 3 as proposed by Becke. [44]
- .NOPRUN Supresses the default pruning of the atomic angular integration grids for radial points close to the nucleus.
- .NORMAL Shortcut keyword for radial integration accuracy 10^{-13} and angular expansion order equal to 35.
- .RADINT
<RADINT>

Determines the quality of the radial integration grid. Default radial integration accuracy is 10^{-11} . Note that the value of `RADINT` is changed by the following keywords: `NORMAL`, `FINE`, `ULTRAFINE`. The `GRIDX` keywords also imply specific `RADINT` values.

- .`ULTRAF` Shortcut keyword for radial integration accuracy 10^{-15} and angular expansion order equal to 64.
- .`LB94` Uses the van Leeuwen-Baerends asymptotic correction. This correction can be combined with all GGA type functionals
- .`CS00` Using the Casida-Salahub asymptotic correction[47, 48]

$$V'_{XC}(\mathbf{r}) = \max(V_{XC} + \Delta, V_{XC} + (1 - \text{HFexchange}) \cdot V_{LB94,correction}), \quad (4.1)$$

with a Zhan-Nichols-Dixon shift [47, 49] (`HFexchange` = 0.2 for B3LYP)

$$\Delta = ZND_1 \cdot \epsilon_{HOMO} - ZND_2 \quad (4.2)$$

Where $ZND_1 = 0.2332$ and $ZND_2 = 0.0116$. Note that $ZND_2 = 0.0116au. = 0.315eV$. The parameters have been optimized with respect to a B3LYP functional so these parameters can be changed using the following keywords. Note that this keyword only works with GGA type functionals. The `CS00` keyword only work with a method which calculates the HOMO energy.

- .`CS00 ZND1`
`<ZND1 in au.>`
 Changes the parameter ZND_1 (default $ZND_1 = 0.2332$) see `.CS00`
- .`CS00 ZND2`
`<ZND2 in au.>`
 Changes the parameter ZND_2 (default $ZND_2 = 0.0116$) see `.CS00`
- .`CS00 SHIFT`
`<Delta in au.>`
 Using the Casida-Salahub correction with a shift of Delta see `.CS00`

4.4.2 *DENSOPT

This section contains the different options for obtaining the SCF wave function and energy. The default is augmented Roothaan–Hall density optimization in combination with a conjugate residual optimal trialvectors (CROP) scheme for solving the (level-shifted) Newton equations ((see ref. [50])) and using an atom-in-a-molecule starting guess.

- `.2ND_ALL` Use second order optimization in all SCF iterations.
- `.2ND_LOC` Use second order optimization in local SCF iterations.
- `.ARH` Use Augmented Roothaan-Hall scheme for density optimization (default) [51, 52] where the CROP solver ([50]) is used for solving the (level-shifted) Newton equation.
- `.ARH FULL` Use Augmented Roothaan-Hall for density optimization - no truncation of the reduced space, keep all micro vectors for the CROP algorithm.
- `.ARH DAVID` Use Augmented Roothaan-Hall for density optimization where the (level-shifted) Newton equations are solved using a reduced space solver based Davidson's algorithm [53]. This setting may be used if default settings converges to a saddle point.
- `.ARH(LS) DAVID` The same procedure as described for `.ARH DAVID` augmented with a line search to accelerate convergence for very large molecular systems, but is less efficient than `.ARH DAVID` for smaller molecular systems.
- `.STABILITY` Check stability of the optimized wave function by calculation of the lowest Hessian eigenvalue.
- `.STAB MAXIT`
 - `<Max. number of stability iterations>`
 - Max. number of iterations in calculation of lowest Hessian eigenvalue (default is 40).
- `.CHOLESKY` Do Cholesky decomposition of overlap matrix (for density optimization in orthogonal AO basis) - default is Löwdin decomposition by diagonalization.
- `.CONFAC`
 - `<Contraction factor>`
 - For update of trust-radius (used in ARH and second order optimization). If trust-radius should be contracted, contract it by this factor (default is 0.7).
- `.CONTRAC`
 - `<Contraction criterion>`
 - For update of trust-radius (used in ARH and second order optimization). Contract trust-radius if trust-radius ratio is smaller than this criterion (default is 0.25).
- `.CONVDYN`
 - `<Option>`
 - Dynamic SCF convergence threshold. This is suitable for calculations on large molecules, since the standard SCF convergence threshold is based on the Frobenius norm of the SCF gradient, which is not size-extensive. Options are SLOPPY, STANDARD,

TIGHT, VTIGHT corresponding to 10^{-4} , 10^{-5} , 10^{-6} , 10^{-7} times the square root of the number of electrons in the system.

.CONVTHR

<Threshold>

SCF convergence threshold - Frobenius norm of SCF gradient (default is 1.0d-4). Note that this convergence criterion is not size-extensive. For large molecules, it is recommended to use .CONVDYN instead.

.DIIS Pulay's DIIS scheme is used to speed up SCF convergence [54, 55].

.DISK Store densities and Fock/KS matrices from previous iterations on disk instead of in core when constructing the new Fock/KS matrix (*memory-saving option*).

.DISKSOLVER Store trial and sigma vectors on disk instead of in core when solving linear equations (*memory-saving option*). Only referenced using ARH , Direct Density optimization or second order optimization.

.EXPAND

<Expansion criterion>

For update of trust-radius (used in ARH and second order optimization). Expand trust-radius if trust-radius ratio is larger than this criterion (default is 0.75).

.EXPFAC

<Expansion factor>

For update of trust-radius (used in ARH and second order optimization). If trust-radius should be expanded, expand it by this factor (default is 1.2).

.FIXSHIFT

<Shift>

Use a fixed level shift in all SCF iterations.

.HESVEC

<Number of Hessian eigenvalues>

Number of lowest Hessian eigenvalues to be calculated with keyword .STABILITY (default is 1).

.LCV Compute the Least-Change Valence orbitals after valence density optimization step. All subsequent calculations are computed using Least-Change Valence orbitals augmented with atomic virtual orbitals of the Atomic density as basis set. Keyword only effective with TRILEVEL starting guess.

- .LCM** Compute the Least-Change Molecular orbitals after the Full molecular density calculation. All subsequent calculations are computed using these orbitals as basis set. **.LCV** is automatically included. Keyword only meaningful with **TRILEVEL** starting guess.
- .LEVELSH**
<N> <shift1> <shift2> ...<shiftN>
Use custom level shifts in the first *N* SCF iterations.
- .LOW ACCURACY START** Use low accuracy settings in the first couple of iterations until the appropriate gradient threshold have been reached.
- .MAXELM**
<Trust radius (max. element)>
Absolute max. element of step allowed (default is 0.35). Used in ARH and second order optimization.
- .MAXIT**
<Max. number of iterations>
Max. number of SCF iterations (default is 100).
- .MAXSTEP**
<Trust radius (Frobenius norm)>
Absolute max. Frobenius norm of step allowed (default is 0.6). Used in ARH and second order optimization.
- .MICTHRS**
<Threshold for micro iterations>
The micro iterations of the CROP algorithm are converged to gradient norm times this factor (default is 1.0d-2). Only referenced if ARH, Direct Density optimization or second order optimization.
- .MICROVECS**
<Max. number of microvectors>
Max. number of microvectors to be kept in the CROP scheme (default is 2). Only referenced if ARH, Direct Density optimization or second order optimization.
- .MINDAMP**
<Minimum damping>
Never allow level shift to be smaller than this.
- .NOAV** Turn off Fock matrix averaging in SCF iterations (this is default since the default optimization scheme is ARH, which contains implicit averaging).

.NVEC

<Max. no of vectors for averaging>

Maximum number of previous Fock and density matrices to be stored and used for ARH and DIIS (default is 10 for HF and 7 for DFT).

.NOPREC Turn off preconditioning of linear equations (ARH, TrFD, and second order optimization).

.NOSHIFT Do no level shifting.

.RESTART The code automatically saves the density matrix in each SCF iteration in a file called `dens.restart`. This option restart the calculation from this density matrix. If the program cannot find the `dens.restart` file the keyword have no effect.

.SKIPSCFLOOP The code automatically saves the density matrix and the Fock/Kohn-Sham matrix at the end of the SCF iterations in two files called `dens.restart` and `fock.restart`. This option reads the files and skips the SCF iterations.

.RH Use standard Roothaan-Hall scheme (diagonalization) for density optimization.

.SOEO Use second order ensemble optimization. Remember to specify active space and starting occupations. Use only in combination with **.NOGCBASIS**. Requires **.HF** or **.DFT** as wave function. After an ensemble optimization, the orbitals are written to file `cmo.out` and the AO density matrix is written to `dao.out`, which may be used to plot molecular orbitals and charge density plots.

.SOEOSPACE

<core> <act>

Specification of the orbitals in the active space. <core> being the number of core orbitals (occupations not optimized), and <act> the number of active orbitals, where the orbital occupations are optimized.

.SOE00CC

<fully occupied>

<fractionally occupied>

<n1> <n2> ...

Specification of the distribution of electron pairs in the starting orbitals in a second order ensemble optimization. <fully occupied> is the number of fully occupied orbitals and <fractionally occupied> is the number of fractionally occupied orbitals. <n1>, <n2> etc. are the fractional occupations of the fractionally occupied spin-orbitals in the starting guess. The given occupations must lie between zero and one. If they are either zero or one, this orbital occupation is not optimized. Equal

distribution may result in wrong results due to symmetry. The sum of the fully occupied orbitals and the given orbital occupations $\langle n1 \rangle$, $\langle n2 \rangle$ etc., must equal the total number of electron pairs in the system. (NOTE that it is possible to specify an active space of zero orbitals, but still include fractional occupations. The optimization will thus optimize the shapes of the orbitals, without optimizing the orbital occupations.)

- .SOEOSAVE** Saves the AO Fock matrix (and the AO density matrix) to the file `soeosave.out`. Used for restart.
- .SOEORST** Restart second order ensemble optimization from orbitals stored in `soeosave.out` (so remember to copy this file to the working directory). The molecular geometry need not be the same as in the calculation where the stored orbitals were created. The active space and starting occupations still need to be specified using `.SOEOSPACE` and `.SOEOOCC`.
- .SOEOGC** Grand-canonical ensemble optimization. Removes the restriction on the total number of electrons from the second order ensemble optimization. The sum of the starting occupations still must be correct.
- .SOEOMATHR**
`<macro threshold>`
Specifies the convergence threshold for the macro iterations in second order ensemble optimization.
- .SOEOMITHR**
`<micro threshold>`
Specifies the convergence threshold for the micro iterations in second order ensemble optimization.
- .SOEOTRUST**
`<trust-radius>`
Specifies the starting trust-radius in the ensemble optimization. Trust-radius will still be updated throughout the optimization.
- .SOEODIPOLE** Use the ensemble density matrix to determine electric dipole moment and static electric polarizability.
- .START**
`<Option>`
Starting guess for SCF optimization. Options are:
- **H1DIAG** Obtain initial guess by diagonalizing one-electron Hamiltonian

- **ATOMS** Obtain initial guess by atoms-in-molecules approach (default)
- **TRILEVEL** Optimization in three steps. 1. Atomic densities, 2. Valence molecular density, 3. Full molecular density [4, 5].

Note that Huckel guess is not supported by LSDALTON.

.VanLenthe Use Van Lenthe's scheme for level shifting and averaging [56].

4.4.3 \$INFO

For the different parts of the wave function optimization, it is possible to get very detailed information. This section describes these keywords. They must be put under ***DENSOPT** in a section beginning with **\$INFO** and ending with **\$END INFO**. Note that these keywords do NOT start with a dot!

INFO_CROP Detailed info from Conjugated Residual Optimal vectors scheme (Direct Density Optimization, ARH and second order optimization).

INFO_DIIS Detailed info from Direct Inversion in the Iterative Subspace algorithm.

INFO_LEVELSHIFT Detailed info from determination of dynamic level shift (Direct Density Optimization, ARH and second order optimization).

INFO_LINEQ Detailed info from solution of linear equations and trust-radius update (Direct Density Optimization, ARH and second order optimization).

INFO_RH Detailed info from Roothaan-Hall algorithm (diagonalization).

INFO_RH_DETAIL Even more detailed info from Roothaan-Hall algorithm (diagonalization).

INFO_STABILITY Detailed info from calculation of lowest Hessian eigenvalue (stability analysis) and HOMO-LUMO gap.

INFO_STABILITY_REDSpace Reduced space info from calculation of lowest Hessian eigenvalue (stability analysis) and HOMO-LUMO gap.

4.5 **OPTIMIZE

This input module describes the keywords needed to optimize a molecular geometry. In the current release no second-order algorithms are available since the molecular Hessian is not implemented, but a number of quasi-Newton trust-radius level-shifted techniques for finding equilibrium geometries are available (**.BFGS**, **.PSB**, **.DFP**). The methods differ in the way approximate Hessians are built and updated: **.INIMOD**, **.INIRED** and **.INITEV**

provide different approximations for the initial Hessian, while `.MODHES` recomputes the approximated Hessian at every geometry.

The geometry optimization generates a sequence of geometry steps. The coordinates of each accepted step are stored in the files `MOLECULE.XXX`, where `XXX` is the geometry step number, and the final converged geometry in `MOLECULE.OUT`.

Geometric structure can be optimized in redundant internal (`.REDINT`) or Cartesian (`.CARTES`) coordinates with two sets of convergence criteria (two for the gradient and two for the step). Convergence is obtained when the following four quantities are respectively smaller than the four convergence criteria ϵ , 5ϵ , 3ϵ and 15ϵ [57]:

- the root-mean-square of the gradient,
- the maximum absolute value of the gradient,
- the root-mean square of the step vector,
- and the maximum absolute element (in internal or Cartesian coordinates) of the step vector.

The default setting specifies the minimization in internal coordinates [58], with a model Hessian [59] updated with BFGS formula. The atoms-in-molecule approach is used to set up the initial density-matrix at each new geometry step. However, it is possible to take as starting guess McWeeny-purified converged density-matrix at the previous geometry. This is achieved by setting the `.RESTART` option under the `*DENSOPT` subsection. The default value of ϵ is 10^{-5} (Other predefined values for the convergence criteria can be set using the keywords `.VLOOSE`, `.LOOSE`, `.TIGHT` and `.VTIGHT`). Alternatively the convergence criteria of Baker [60] can be set using `.BAKER`.

Example of test cases for geometry optimization calculations

The test cases are located in `LSDALTON/test/geomopt`. These test cases illustrate the main features of the geometry optimization:

- `geomopt/cartes_bfgs_min`: performs a DFT/LDA geometry optimization in Cartesian coordinates, with BFGS update of an initial Hessian equal to a unit matrix and Baker convergence criteria.
- `geomopt/redint_bfgs`: performs a DFT/BLYP geometry optimization in redundant internal coordinates with a model Hessian updated with BFGS formula and Baker convergence criteria.
- `geomopt/Excited_state_opt`: performs a DFT/B3LYP geometry optimization of an excited state in redundant internal coordinates and Baker convergence criteria.

- *geomopt/cartes_psb_min*: performs a DFT/BLYP geometry optimization in Cartesian coordinates with a diagonal initial Hessian updated by PSB formula, McWeenie's purification used for constructing the initial guess and Baker convergence criteria.
- *geomopt/geomopt_constrain*: shows a constrained geometry optimization of ethane, keeping the C-C bond distance fixed (Internal coordinate #1 as determined from the use of `.FINDRE`).
- *geomopt/cartes_trilevel_newconv*: performs a geometry optimization in Cartesian coordinates with `.TRILEVEL` starting guess and a default convergence criteria, a diagonal initial Hessian updated by a loose DFP formula.

`.BAKER` Activates the convergence criteria of Baker [60]. The minimum is then said to be found when the largest element of the gradient vector (in Cartesian or redundant internal coordinates) falls below $3 \cdot 10^{-4}$ and either the energy change from the last iteration is less than 10^{-6} or the largest element of the predicted step vector is less than $3 \cdot 10^{-4}$.

`.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.

`.CARTES` Specifies that Cartesian coordinates should be used in the optimization.

`.CONSTR`

<Number of constrained redundant internal coordinates>

<Indices of each constrained redundant internal coordinate, one per line>

Specifies a constrained geometry optimization. You will need to give the total number of redundant internal coordinates to constrain and their indices as defined using the `.FINDRE` keyword. For example fixing redundant internal coordinates 4 and 9 would read like:

...

****OPTIMIZE**

`.CONSTR`

2

4

9

...

****END OF INPUT**

- `.DFP` Specifies that a first-order method with the Davidon-Fletcher-Powell (DFP) update formula should be used for optimization.
- `.FINDRE` Using this keyword no geometry optimization is carried out, but only the redundant internal coordinates are found.
- `.INIMOD` Use a simple model Hessian [59] 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 optimizations in redundant internal coordinates. Currently this is not an option for optimization in Cartesian coordinates.
- `.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.
- `.INITEV`
<Read EVLINI, the eigenvalues>
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.
- `.LOOSE` Default convergence criterion ϵ set to $1.0 \cdot 10^{-4}$.
- `.MAX IT` Maximum number of iterations in geometry optimization scheme (default set to 100).
- `.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.
- `.PRINT`
<Print level>
Print level for the output. The higher the print level, the more information is printed to the output file.
- `.REDINT` Specifies that redundant internal coordinates should be used in the optimization. This is the default.

.PSB Specifies that a first-order method with the Powell-Symmetric-Broyden (PSB) update formula should be used for optimization.

.TIGHT Default convergence criterion ϵ set to $1.0 \cdot 10^{-6}$.

.VLOOSE Default convergence criterion ϵ set to $1.0 \cdot 10^{-3}$.

.VTIGHT Default convergence criterion ϵ set to $1.0 \cdot 10^{-7}$.

4.6 **DYNAMI

This input module is devoted to molecular dynamics simulations. LSDALTON provides some tools for performing direct Born-Oppenheimer molecular dynamics, i.e. trajectories are integrated with forces calculated on the fly. Trajectory integration is available for SCF-type wavefunctions (with dispersion correction for DFT) and for DEC wavefunctions. Properties could be calculated along the trajectories and should be specified in the **RESPONSE section. LSDALTON simulates microcanonical(NVE) ensembles. Currently no sampling is available and the starting coordinates and velocities must be provided as input. Trajectories are integrated with gradient-based velocity-Verlet method [61].

LSDALTON provides several options for the choice of starting guess for SCF-type electronic theory methods. By default, the guess is generated anew each time step according to the corresponding input. Alternatively, one can restart from the density optimized at the previous time-step. This is achieved by setting the .RESTART option under the *DENSOPT subsection. This reduces the number of SCF iterations per time-step and accelerates the integration. Two more efficient algorithms for starting guess propagation are implemented in LSDALTON Fock matrix dynamics [62] and time-reversible propagator [63].

Example test cases for **DYNAMI calculations:

These test cases illustrate the main features of direct dynamics. The test cases are located in LSDALTON/test/ddynam:

- *ddynam/H2O_fmd*: free water molecule with Fock matrix dynamics;
- *ddynam/HCN_ddyn*: HCN dissociation with restart density;
- *ddynam/HF_vib*: a vibrating HF molecule with time-reversible propagator.

.PRINT

<Print level>

Print level for the output. The higher the print level, the more information is printed to the output file.

.TIMESTEP

Defines integration time step in femtoseconds which should be given in the next line (real). The default is 0.5 fs.

.MAX ITER

Defines maximum number of time steps. Must be given in the next line (integer). The default is 100.

.MASSWE

States that the integration is carried out in mass-weighted coordinates. The default is FALSE.

.MWVEL

States that the input velocities are mass-weighted. The default is FALSE.

.VELOCI

Begins the input of the velocities. In the next line the number of atoms should be given (integer), then the velocities grouped atom-wise: three(x,y,z) in a line (real). The default is for all velocities to be zero.

.FOCKMD

Fock matrix dynamics. Below number of points for extrapolation N and polynomial order M must be specified. Note that $M > N$ (both integer). Works only with .RESTART option under the *DENSOPT subsection

.TIMREV

Time-reversible density propagation. Below the number of points for extrapolation N (integer) must be specified. The only thoroughly tested option is $N = 2$.

4.7 **LOCALIZE ORBITALS

This section describes keywords needed to perform localization of Hartree–Fock orbitals. The optimization of the chosen localization function will be performed using a trust-region algorithm [64], and can be performed for the occupied space and/or the virtual orbital space. For the occupied space the core and valence spaces will be localized separately.

After running a Hartree-Fock calculation followed by an orbital localization two files with molecular orbital coefficients will be present; *cmo_orbitals.u* and *lcm_orbitals.u*. *cmo_orbitals.u* contains coefficients for the canonical molecular orbitals (CMO) and *lcm_orbitals.u*

contains coefficients for the localized molecular orbitals (LMO). The coefficient matrix row index refers to the atomic orbital basis index, while the column index refers to the molecular orbital basis. The format of the files are as described in the Matrix File Format part of Section 3.

For the most efficient orbital localization for large molecular systems [65], the .ARH, .START/TRILEVEL and .LCM options are recommended for use as described in Section 4.4.2, and .PSM with powers 2 2 are recommended for the **LOCALIZE ORBITALS section.

.PSM

<m m>

Use powers m of the second moment (PSM) localization function [66]. A power 0 specifies that no localization should be carried out (e.g., 0 1 to only localize the virtual space). m=1 is equivalent with the Boys localization function. For improved locality m \geq 2 is recommended.

.PFM

<m m>

Use powers m of the fourth moment (PFM) localization function [67]. A power 0 specifies that no localization should be carried out (e.g., 0 1 to only localize the virtual space). Using power 1 yields orbitals with good locality, but a power 2 is recommended if locality of the orbitals is essential. Powers higher than 3 are not recommended. The PFM localization function should be chosen if basis sets augmented with diffuse functions are used.

.PipekMezey

Use Pipek-Mezey localization function using an implementation based on Löwdin population analysis rather than the traditional Mulliken population analysis (see Ref. [68]). Both occupied and virtual orbitals will be localized. Please note, the Pipek-Mezey localization function suffers from system and basis set dependencies. For systems with complicated bonding structures or when using basis sets augmented with diffuse functions the PSM or PFM options should be used if good locality is essential.

.OccPipekMezey

Pipek-Mezey localization of only the occupied space.

.VirtPipekMezey

Pipek-Mezey localization of only the virtual space.

.PipekM(MULL)

Use Pipek-Mezey localization function using the Mulliken population analysis. A

power 0 specifies that no localization should be carried out. Otherwise the same considerations as for the `.PipekMezey` option.

`.OccPipekM(MULL)`

Pipek-Mezey localization (using Mulliken population analysis) of occupied space.

`.VirtPipekM(MULL)`

Pipek-Mezey localization (using Mulliken population analysis) of virtual space.

`.No Level2 Localization`

Should only be used in combination with the `.START/TRILEVEL` and `.LCM` options for the `*DENSOPT` section. The keyword is used to skip orbital localization after the second level in the three level procedure. Not recommended for general use, but can be used to save computational time if only the virtual space is to be localized.

`.Only Loc`

Provided the file with the molecular orbital coefficients (e.g., `cmo_orbitals.u` or `lcm_orbitals.u`) using this keyword the Hartree-Fock calculation will be skipped, and the orbitals will be read in and localized. Important: rename or copy the orbital coefficient file to `orbitals_in.u`, only this file will be read. After the localization procedure the molecular orbital coefficients corresponding to the localized basis is written to `orbitals_out.u`. Thus, for this keyword to be used, `orbitals_in.u` must exist and a localization function must be chosen.

`.Orbital Locality`

This keyword will make sure second and fourth moment orbital spreads are printed for all orbitals, not just the least local ones.

`.MACRO IT`

<option> Set maximum number of macro iterations for orbital localization procedure. Default is 200.

`.LOOSE MICRO THRESH`

Looser threshold for solving level-shifted Newton equations in davidson solver in orbital localization procedure. Should only be used if a calculation fails and the error message tells you to use this keyword.

`.Orbital Plot`

<orbital specification>

Generates `.plt` file for either the least local or the most local orbitals. To generate `.plt` files for the least local orbitals use orbital specification `LEASTL`, whereas to generate `.plt` files for the most local orbitals use specification `MOSTL`. If both least

local and most local are wanted, make two `.Orbital Plot` sections with each of the specifications. The `.plt` files may be plotted using for example the UCSF Chimera program package [1]. Note: for plotting other orbitals, use the `**PLT` option described in section 4.11. Also note that the grid dimensions may be modified as described in Section 4.12.

4.8 ****RESPONS**

This section describes the keywords needed for obtaining molecular properties for HF and DFT based on the atomic orbital based response formulation [69]. The response section begins with `**RESPONS`. General response information related to more than one property (e.g. how many excitation energies) is put directly under `**RESPONS`. After this, each individual response property (e.g. the polarizability tensor) is labeled by an asterisk, and the specific (optional) information related to that property (e.g. which optical frequencies) follows below. All input values are given in atomic units. The general structure is thus:

```
**RESPONS

.General response information

*Label for property 1

.Specific (optional) information for property 1

*Label for property 2

.Specific (optional) information for property 2
```

Each of the properties below have a test case. The test cases may be found in the **LS-DALTON/test/LSresponse** directory. To run a test case, go to the `test/` directory and type:

```
./TEST <test case>
```

The test case files contains simple examples of input files.

Below, the list of general response input keywords follows.

`.NEXCIT`

`<Number of excitation energies>`

Determines the number of excitation energies to be calculated. If this keyword is set, the excitation energies and the corresponding one-photon dipole absorption strengths for the lowest NEXCIT excited states (i.e. excited states 1 to NEXCIT) are always calculated. **Important:** If properties involving excited states are to be requested

(e.g. two-photon absorption or the excited state gradient), this keyword must be listed *before* any of these properties.

Test case: LSresponse/LSresponse_HF_opa

4.8.1 *SOLVER

General information related to the way response equations are solved to obtain a given property. Please note that for calculations that involve OPENRSP, these settings should be specified inside the **OPENRSP input section as described in Sections 4.9.3 and 4.9.5. By default, standard and eigenvalue response equations are solved using the algorithm with paired trial vectors [70], whereas damped (complex) response equations are solved using the algorithm with symmetrized trial vectors [71].

This choice can be modified using the .SYM_SOLVER and .PAIR_SOLVER keywords, see below.

.SYM_SOLVER

The solver with symmetrized trial vectors is used for solving standard response equations.

Test case: LSresponse/LSresponse_HF_alpha_astv

.PAIR_SOLVER

The solver with paired trial vectors is used for solving damped (complex) response equations.

Test case: LSresponse/LSresponse_DFT_alpha_aptv

.CONVTHR

<Convergence threshold>

Convergence threshold to which response equations are solved. Default value is set to 10^{-4} .

.MAXIT

<Maximum number of iterations>

Maximum number of iterations in the iterative procedure. Default value is set to 100.

.MAXRED

<Maximum size of the reduced space>

Maximum size of the reduced space. Default value set equal to 200.

.AOPREC

AO preconditioning is used (MO preconditioning by default). AO preconditioning may only be used within the algorithm with paired trial vectors.

.NOPREC

No preconditioning is used.

.CONVDYN**<Option>**

Dynamic convergence threshold suitable for large calculations. Options are TIGHT, STANDARD, and SLOPPY.

.RESTEXC**<Number Of excitation Vectors on file>**

This is a restart option, that allows the user to restart the calculation.

Assume for instance that you have already done one calculation where you requested 6 excitation energies but realize that you need 10. You can provide the file `scf_rsp_eigenvecs_ao.restart` and use

.RESTEXC

6

option to restart the calculation from the previous. This option can also be used to bypass the excitation vector step if the previous calculation failed for some property evaluation which needed the excitation vectors.

.DTHR**<Threshold>**

Threshold for when excited states are to be considered degenerate

Now follows a list of input keywords for the specific properties available in LSDALTON.

4.8.2 *DIPOLE

Calculation of the permanent dipole moment

4.8.3 *DIPOLEMOMENTMATRIX

Calculation of the full dipole moment matrix:

$$\mu = \begin{pmatrix} \mu_{00} & \mu_{01} & \mu_{02} & \dots \\ \mu_{10} & \mu_{11} & \mu_{12} & \dots \\ \mu_{20} & \mu_{21} & \mu_{22} & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix} \quad (4.3)$$

- μ_{00} denote the permanent dipole moment for the ground state
- μ_{0i} denote the transition dipole moment between the ground state and the i 'th excited state

- μ_{ii} denote the Permanent dipole moment for the i'th excited states
- μ_{ij} denote the transition dipole moment between the i'th and j'th excited state

Requires specification of .NEXCIT!

4.8.4 *ALPHA

Calculation of the (possibly complex) electric dipole polarizability tensor α_{AB} and its isotropic average. The polarizability equals minus the linear response function $\langle\langle\mu_A; \mu_B\rangle\rangle_{\omega_B}$, where ω_B is in general allowed to be complex – that is, $\omega_B = \omega_B^R + i\omega_B^I$. Unspecified frequencies are set to zero by default, i.e. if no frequencies are specified, the static polarizability is calculated.

Test case: LSresponse/LSresponse_HF_alpha

.BFREQ

<Number of real frequencies for B operator>

<Freq1, freq2, ... , freqN>

Real frequencies ω_B^R in the corresponding linear response function. The first line after .BFREQ contains the number of frequencies and the second line contains all frequency values (on one single line).

.IMBFREQ

<Number of imaginary frequencies for B operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_B^I in the corresponding linear response function. The imaginary part of the complex polarizability describes a broadened one-photon absorption spectrum. If .IMBFREQ is not specified, only the real polarizability is calculated. Input as for .BFREQ.

For example, the following input is used to calculate complex polarizability tensors for the frequencies $\omega_B = 0.1 + 0.05i$ and $\omega_B = 0.2$ (in a.u.):

****RESPONS**

***ALPHA**

.BFREQ

2

0.1 0.2

.IMBFREQ

2

0.05 0.0

4.8.5 *BETA

Calculation of the (possibly complex) first electric dipole hyperpolarizability tensor β_{ABC} and the corresponding averages β_{\parallel} and β_{\perp} . The first hyperpolarizability tensor equals minus the quadratic response function $\langle\langle\mu_A; \mu_B, \mu_C\rangle\rangle_{\omega_B, \omega_C}$, where the frequencies are in general allowed to be complex – i.e. $\omega_B = \omega_B^R + i\omega_B^I$ and $\omega_C = \omega_C^R + i\omega_C^I$. The input is analogous to the *ALPHA input. Unspecified frequencies are set to zero by default, i.e. if no frequencies are specified, the static first hyperpolarizability is calculated.

Test case: LSresponse/LSresponse_HF_beta

.BFREQ <Number of real frequencies for B operator><Freq1, freq2, ... , freqN>Real frequencies ω_B^R in the corresponding quadratic response function. Input as for .BFREQ under *ALPHA.

.IMBFREQ

<Number of imaginary frequencies for B operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_B^I in the corresponding quadratic response function. Input as for .IMBFREQ under *ALPHA.

.CFREQ

<Number of real frequencies for C operator>

<Freq1, freq2, ... , freqN>

Real frequencies ω_C^R in the corresponding quadratic response function. Input as for .BFREQ under *ALPHA.

.IMCFREQ

<Number of imaginary frequencies for C operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_C^I in the corresponding quadratic response function. Input as for .IMBFREQ under *ALPHA.

4.8.6 *DAMPED_TPA

Damped two-photon absorption [72] where both individual photons have the same energy (= half the excitation energy).

Test case: LSresponse/LSresponse_HF_dtpa

.OPFREQ

<Number of one-photon frequencies>

<Freq1, freq2, ... , freqN>

The frequencies in the input are one-photon frequencies (=half the excitation energy in case of resonance).

.GAMMA

<Damping parameter>

The damping parameter γ is used in *all* response equations, i.e. $i\gamma$ is effectively added to all frequencies occurring in the response equations. Default value: 0.005 a.u.

4.8.7 *QUASIMCD

Calculation of the \mathcal{A} and \mathcal{B} terms of magnetic circular dichroism (MCD), based on the method of Ref. [73], but reformulated using the formalism of Ref. [69]. The calculation of Damped MCD spectra according to Ref. [74] is also possible with this keyword.

Note that one or more of the 3 keywords { .DAMPEDXCOOR , .MCDEXCIT, .DAMPEDRANGE } must be specified

If **.MCDEXCIT** is specified, the default is to proceed in the following way:

1. Determine the excited states requested under **.MCDEXCIT**
2. Determine \mathcal{B} terms for all allowed transitions. Both London atomic orbitals LAO (also called Gauge including atomic orbitals GIAO) and non-LAOs are used to determine the \mathcal{B} terms. LAOs are superior to standard AOs, but the difference can give an indication of how close the result is to the basis set limit.
3. Simulate the MCD spectrum based on the calculated \mathcal{B} terms (sticks) by associating a Lorentzian lineshape function (with lineshape parameter of 0.005 a.u.) to each \mathcal{B} term (See Ref. [73]). The simulated spectrum is written to a file MCDspectraAU.dat (in atomic units). The first number is the frequency, the second the LAO ellipticity and the third is the non-LAO ellipticity. A corresponding file is generated called MCDspectra.dat using standard units (cm^{-1} for frequencies and molar ellipticity for the ellipticity). The individual \mathcal{B} term contribution to the MCD spectra is written to the files BtermAU.dat and Bterm.dat. Per default 5000 points are used for the simulated spectrum. The standard units for the \mathcal{B} term are given as $\frac{D^2\mu_B}{cm^{-1}}$, where D is the unit Debye and μ_B is the Bohr magneton. The conversion factor between atomic units and the standard units is 5.8876410^{-5} . The standard unit for the \mathcal{A} term is given as $D^2\mu_B$, with a conversion factor of 12.920947806163007.
4. Perform damped calculations based on the standard MCD calculation. A number of points around each individual MCD peak (default value: 10) are determined for which

to calculate the damped spectra. The result is printed to the files dampedMCDspectraRAU.dat (atomic units) and dampedMCDspectraR.dat.

Test case: LSresponse/LSresponse_mcd_calc, LSresponse/LSresponse_mcd
LSresponse/LSresponse_mcd2, LSresponse/LSresponse_mcd3

Note that only the LSresponse/LSresponse_mcd_calc test case is a good example for a general MCD calculation.

.MCDEXCIT

<Number of excited states to consider>

The \mathcal{B} terms will be calculated for all states, unless they are zero due to selection rules (dipole forbidden)

.MCDEXSTATES

<Number of specific excited states to consider in MCD calc>

<Specific excited states to consider>

This keyword only calculates the \mathcal{B} and possible \mathcal{A} terms for the selected excited states. The first line after .MCDEXSTATES contains the number of specific excited states for which the MCD is to be calculated, and the second line specifies such states (identified by their index in the .MCDNEXCIT list). **Important:** The position of the highest lying excited state specified by .MCDEXSTATES must be smaller than or equal to the total number of excited states specified by .MCDNEXCIT!

.DEGENERATE

This keyword must be set in input if the molecule has a degenerate state so that \mathcal{A} terms are possible, otherwise all states are assumed non-degenerate and no \mathcal{A} terms will be calculated. When this keyword is set, the \mathcal{A} terms are calculated and a derivative Lorentzian lineshape function is used to simulate the individual \mathcal{A} term contributions to the MCD spectrum (See Ref. [73]). The individual \mathcal{A} terms contributions to the MCD spectrum are written to the files AtermAU.dat and Aterm.dat

.DAMPEDXCOOR

<Number of frequencies>

<Freq1, freq2, ... , freqN>

The frequencies for which to calculate the damped MCD spectra (in atomic units).

.DAMPEDRANGE

<Freq1>

<FreqN>

<Number of frequencies in range between Freq1 and Freq2>

The frequencies for which to calculate the damped MCD spectra (in atomic units), here specified as an interval.

- .NO LONDON**
Deactivates the use of LAOs (i.e., only standard, non-LAO, basis is used)
- .NO NONLONDON**
Deactivates the use of non-LAOs (i.e., only LAO are used)
- .NO SIMULATE**
Deactivates the simulation of the MCD spectrum from the individual MCD terms
- .NO ATERM**
Deactivates the calculation of \mathcal{A} terms
- .NO BTERM**
Deactivates the calculation of \mathcal{B} terms
- .NO DAMPED**
Deactivates the execution of damped MCD spectra
- .GAUSSIAN**
Uses Gaussian lineshape functions (with a lineshape parameter of 0.0070851079363103793) instead of the default Lorentzian lineshape functions.
- .LINESHAPEPARAM**
<lineshape parameter>
Changes the lineshape parameter (default value is 0.005 a.u. for Lorentzian, and 0.0070851079363103793 a.u. for Gaussian) used to simulate the spectrum
- .NSTEPS**
<number of frequencies used for the simulated MCD spectra>
Changes the number of frequencies used for the simulation of the MCD spectra from the individual \mathcal{A} and \mathcal{B} terms' sticks (default 5000).
- .NVECFORPEAK**
<number of frequencies used around each MCD peak>
Changes the number of frequencies used around each MCD peak (default 10) in the damped response calculation.

Note that the keyword **.DTHR** under ***SOLVER** (see section 4.8.1) can be used to define the threshold below which an excited state is considered degenerate

4.8.8 *SHIELD

Nuclear magnetic shielding tensors and chemical shift can be calculated with this keyword

Test case: LSresponse/LSresponse_NMR_SHIELD

4.8.9 *INASHIELD

Individual atom nuclear magnetic shielding tensors and chemical shifts. NB: Requires the use of SubSystem labels in the MOLECULE.INP file see testcase **Test case:** LSresponse/LSresponse_NMR_IANS_SHIELD_NoOpenRSP

ESDIPOLE

Permanent dipole moment for excited states. **Always requires specification of .NEXCIT!**

If .EXSTATES is not specified, the excited state dipole moments for all excited states from 1 to NEXCIT are calculated.

Test case: LSresponse/LSresponse_HF_esd

.EXSTATES

<Number of specific excited states to consider>

<Specific excited states to consider>

Only calculate excited state gradient for selected excited states. The first line after .EXSTATES contains the number of excited states for which the excited state dipole moment is to be calculated, and the second line specifies such states (identified by their index in the .NEXCIT list). **Important:** The position of the highest lying excited state specified by .EXSTATES must be smaller than or equal to the total number of excited states specified by .NEXCIT!

4.8.10 *ESGRAD

Calculation of the molecular gradient for excited states. **Always requires specification of .NEXCIT!** If .EXSTATES is not specified, excited state gradients for all excited states from 1 to NEXCIT are calculated.

Test case: LSresponse/LSresponse_HF_esg

.EXSTATES

<Number of specific excited states to consider>

<Specific excited states to consider>

Input is identical to .EXSTATES under *ESDIPOLE.

For example, the following input is used to calculate excited state gradients for excited states number 3 and 6:

```
**RESPONS
```

```
.NEXCIT
```


6

*ESGRAD

.EXSTATES

2

3 6

4.8.11 *GAMMA

Calculation of the (possibly complex) second electric-dipole hyperpolarizability tensor γ_{ABCD} and the corresponding averages γ_{\parallel} and γ_{\perp} . The second hyperpolarizability tensor equals minus the cubic response function $\langle\langle\mu_A; \mu_B, \mu_C, \mu_D\rangle\rangle_{\omega_B, \omega_C, \omega_D}$, where the frequencies are in general allowed to be complex – i.e. $\omega_B = \omega_B^R + i\omega_B^I$, $\omega_C = \omega_C^R + i\omega_C^I$, and $\omega_D = \omega_D^R + i\omega_D^I$. The input is analogous to those for *ALPHA and *BETA. Unspecified frequencies are set to zero by default, i.e. if no frequencies are specified the static second hyperpolarizability is calculated.

Test case: LSresponse/LSresponse_HF_gamma

.BFREQ

<Number of real frequencies for B operator>

<Freq1, freq2, ... , freqN>

Real frequencies ω_B^R in the corresponding cubic response function. Input as for .BFREQ under *ALPHA.

.IMBFREQ

<Number of imaginary frequencies for B operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_B^I in the corresponding cubic response function. Input as for .BFREQ under *ALPHA.

.CFREQ

<Number of real frequencies for C operator>

<Freq1, freq2, ... , freqN>

Real frequencies ω_C^R in the corresponding cubic response function. Input as for .BFREQ under *ALPHA.

.IMCFREQ

<Number of imaginary frequencies for C operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_C^I in the corresponding cubic response function. Input as for .IMBFREQ under *ALPHA.

.DFREQ

<Number of real frequencies for D operator>

<Freq1, freq2, ... , freqN>

Real frequencies ω_D^R in the corresponding cubic response function. Input as for .BFREQ under *ALPHA.

.IMDFREQ

<Number of imaginary frequencies for D operator>

<Freq1, freq2, ... , freqN>

Imaginary frequencies ω_D^I in the corresponding cubic response function. Input as for .IMBFREQ under *ALPHA.

4.8.12 *MOLGRA

Single point calculation of the molecular gradient.

Test case: LSresponse/LSresponse_HF_molgra

4.8.13 *TPA

Two-photon absorption where both individual photons have the same energy (= half the excitation energy). **Always requires specification of .NEXCIT!** If .EXSTATES is not specified, two-photon absorption for all excited states from 1 to NEXCIT are calculated.

Test case: LSresponse/LSresponse_HF_tpa

.EXSTATES

<Number of specific excited states to consider>

<Specific excited states to consider>

Input is identical to .EXSTATES under *ESDIPOLE.

4.8.14 *NTO

Generates grid files to plot natural transition orbitals (NTOs) corresponding to the calculated transitions. See e.g. Ref. [75, 76] for a definition of NTOs and for the explicit working equations used in our implementation.

A given NTO pair will produce two files, `nto_occYYYY_excXXXX.plt` and `nto_virYYYY_excXXXX.plt` for the occupied and virtual NTOs respectively. YYYY corresponds to the index of the orbital in the full orbital space, while XXXX is the index of the transition. The grid files can be visualized e.g. using the Chimera program [1].

.THRES

<Squared singular value threshold. Default is 0.1>

A plt file will be generated for all NTOs with a squared singular value higher than THRES.

Test case: LSresponse/LSresponse_DFT_NTO

4.9 **OPENRSP

4.9.1 Disclaimer

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**:

- The functionality is **not fully tested**, although a number of tests are operational.
- The only level of theory currently available is **Hartree-Fock**. The support for DFT level calculations of a large collection of properties is at a late stage of development.
- Calculations can not make full use of MPI parallelism although single-node parallelism for perturbed integral contributions is available.
- The current release support GNU compilers versions 4.9.2 to 6.4.2, and OpenMPI versions 1.8.4 to 2.1.2. Note that certain compiler setups — notably Intel compilers — produce executables that do **not run correctly**.

Work to address these and other issues is ongoing, and the resulting version will be made available in a patch to be released separately. In the meantime, **we strongly discourage** to use the OPENRSP functionality for production calculations. The present functionality is offered to allow users to get acquainted with the running of OPENRSP while the patch version is being finalized. Please also note that the features described in this section may not work as intended and the occurrence of passing tests of the OPENRSP functionality does not guarantee that the corresponding calculation setup and results are without errors.

The OPENRSP functionality is **not enabled by default** when installing LSDALTON, but can be manually activated by adding the `-DENABLE_OPENRSP=ON` flag during setup.

4.9.2 Citation guide

All published results obtained with the OPENRSP functionality are expected to cite the following references:

- Thorvaldsen, A. J., Ruud, K., Kristensen, K., Jørgensen, P., Coriani, S., *J. Chem. Phys.* **129**, 214108 (2008)
- Ringholm, M., Jonsson, D., Ruud, K., *J. Comput. Chem.* **35**, 622 (2014)
- OpenRSP, an open-ended response property library, www.openrsp.org

However, please note that OPENRSP calculations intended for publication **should not be carried out** in the present version, as described in Section 4.9.1, and this citation guide is therefore currently included only for informational purposes.

4.9.3 Introduction

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**. Please read Section 4.9.1 before proceeding.

This section describes the keywords needed for obtaining molecular properties using the OPENRSP library. The functionality described here can be used to calculate response properties for geometric and electric dipole perturbations. Calculations must currently be done at the Hartree-Fock level and the user must make sure that LSDALTON is not set to run at other levels of theory.

OPENRSP can calculate several response properties in one invocation in order to utilize intermediate results that may be common between several properties. An example of a collection of properties where this may be fruitful is the collection of the tensor associated with the molecular polarizability and the corresponding tensor for the first hyperpolarizability, where perturbed Fock, density and overlap matrices determined during the calculation of one of the tensors may potentially be used in the calculation of the other.

The keyword ****OPENRSP** signifies the beginning of the input section for calculations using OPENRSP, and this section has the following general structure:

```
**OPENRSP

*SOLVER (optional)

Response solver flags (optional)

*RESPONSE FUNCTION

Optional flags

$PROPERTY

Information for property 1
```

\$PROPERTY

Information for property 2

...

Since OPENRSP strictly returns response properties (using the keyword `*RESPONSE FUNCTION`), the user must perform any transformation that may be needed to generate the property of interest. For example, the molecular polarizability $\alpha_{\alpha\beta}(-\omega; \omega)$ is defined as the negative of the corresponding response property, i.e. $\alpha_{\alpha\beta}(-\omega; \omega) = -\langle\langle \hat{\mu}_\alpha; \hat{\mu}_\beta \rangle\rangle_\omega$, and the response tensor returned by OPENRSP must therefore be multiplied by -1 to obtain the polarizability tensor.

4.9.4 General settings

Calculations using OPENRSP should include the following input options

`**WAVE FUNCTIONS`

`.HF`

`*DENSOPT`

`.CONVTHR`

`1.0e-8`

`**GENERAL`

`.NOGCBASIS`

`**INTEGRAL`

`.NO SCREEN`

`.NOFAMILY`

In the current release, OPENRSP functionality is limited to HF only. We recommend a tight convergence of the HF wavefunction, and deactivation of Family- and GC-basis functionality is required. Furthermore, the integral screening has not been thoroughly tested for the integral components needed for OPENRSP calculations, and we recommend to deactivate screening. Finally note, that the performance of the integral components is far from optimal. This will be improved in coming updates to the code.

4.9.5 *SOLVER

This is an optional section to set various options associated with the solution of response equations when such solution is requested by OPENRSP. The input structure for this section is identical to the one described in Section 4.8.1 but please be advised that not all of the keywords described there may be valid for use with OPENRSP. However, using the .CONVTHR keyword to set the equation solution convergence threshold is valid and this is typically the most relevant keyword for use with OPENRSP. Note that if a *SOLVER section is entered outside the **OPENRSP keyword section of the input file, this may result in the settings entered in that *SOLVER section not being passed on to OPENRSP. There is one additional keyword for the *SOLVER section of OPENRSP

.PRINT Print convergence information from the response solver to LSDALTON.OUT

Default settings will be used if the *SOLVER section is omitted. Please note that the default solution convergence threshold is 10^{-4} and that this may be insufficient for satisfactory precision for high-order response properties.

4.9.6 *RESPONSE FUNCTION

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**. Please read Section 4.9.1 before proceeding.

Molecular response functions can be written as derivatives of a time-averaged quasienergy Lagrangian $\{L(t)\}_T$:

$$\langle\langle\hat{A};\hat{B}_1,\dots,\hat{B}_M\rangle\rangle_{\omega_{B_1},\dots,\omega_{B_M}} = \frac{d^{M+1}\{L(t)\}_T}{d\varepsilon_{\omega_A}d\varepsilon_{\omega_{B_1}}\cdots d\varepsilon_{\omega_{B_M}}}\bigg|_{\{\varepsilon\}=0}, \quad (4.4)$$

$$\omega_A = -\sum_{j=1}^M \omega_{B_j}, \quad (4.5)$$

where $\hat{A}, \hat{B}_1, \dots, \hat{B}_M$ are perturbing operators and where $\omega_A, \omega_{B_1}, \dots, \omega_{B_M}$ are their associated frequencies. We note that in addition to being able to calculate several different properties in one run, OPENRSP can also in one invocation calculate a given property for several frequency configurations $(\omega_A, \omega_{B_1}, \dots, \omega_{B_M})$.

As indicated in Section 4.9.3, each property is headed by the \$PROPERTY keyword, and there is another input section following this keyword, i.e. the “Information for property i” as shown in Section 4.9.3, where i is a given property. This input section has a general structure as follows:

\$PROPERTY

.PERTURBATION

```

<Number of perturbations>
<Label of perturbation 1>
<Label of perturbation 2>
<...>

```

Specifies the number of perturbations $N \equiv M + 1$, i.e. the order of energy derivative to be calculated, and the perturbation labels describing each perturbing operator in Eq. (4.4). Supported labels are

- GEO (geometric perturbation - Cartesian displacements) and
- EL (electric dipole perturbation).

.FREQUENCY

```

<Number of frequency configurations>
<1st frequency to be specified for configuration 1>
<2nd frequency to be specified for configuration 1>
<...>
<1st frequency to be specified for configuration 2>
<2nd frequency to be specified for configuration 2>
<...>

```

Specifies first the the number of different frequency configurations with respect to which the property in question is to be calculated, i.e. the number of different choices of frequencies associated with the perturbing operators, and then all relevant frequency values, i.e. all of the values of $\omega_A, \omega_{B_1}, \omega_{B_2}, \dots$ that should be specified for each frequency configuration, in atomic units. Which frequencies that should be specified is explained below. The **.FREQUENCY** keyword section may be omitted entirely. In this case, all frequencies for the perturbations associated with the property in question will be set to zero by default. If the **.FREQUENCY** keyword section is used, then the following conditions apply:

- The condition given by Eq. 4.5 means that the frequency of one of the perturbations depends on the sum of the frequencies of all the other perturbations.
- All GEO perturbations are assumed to have a frequency of zero associated with them, but frequencies of EL perturbations may take any value.
- Let M_{EL} be the number of EL perturbations for the property in question. Since the present version only supports GEO or EL perturbations, the two preceding points therefore imply that $M_{\text{EL}} - 1$ frequencies must be specified to correctly specify a frequency configuration of the property in question: The frequency of the remaining EL perturbation will be set equal to the negative sum of the frequencies of all the other EL perturbations.

- All necessary frequencies for one frequency configuration must be given before moving to the next frequency configuration (starting immediately on the next line).

The examples in Section 4.9.7 may prove helpful in demonstrating how these conditions are correctly applied.

.KN-RULE

<Choice k of (k,n) rule>

Specifies the number k for the (k,n) calculation rule choice, where k may be chosen so that $0 \leq k \leq \lfloor (N - 1)/2 \rfloor$ (where $(N - 1)/2$ is rounded down for even N), where N is the total number of perturbations defining the property. OPENRSP will then itself determine the corresponding value of n . A detailed explanation of the meaning and significance of the (k,n) rule choice can be found in the literature[77, 69] and the user should note that the choice of rule can have a dramatic impact on the required calculation time. A quick summary is that the choice $k = 0$ is equivalent to the so-called $n + 1$ rule while the maximum allowed choice of k is equivalent to the so-called $2n + 1$ rule, while other choices (if possible at a given order) are intermediate between these two. For users not seeking a detailed explanation of the impact and significance of the various rule choices, the following simplified guidelines will in the vast majority of cases provide a good choice of calculation rule:

- For properties containing only EL (electric dipole) or only GEO (geometric) perturbations, choose the maximum allowed value of k , i.e. $(N - 1)/2$ (rounded down for even N).
- For properties containing both GEO and EL perturbations, choose k so that $k = N_{\text{GEO}} - 1$, where N_{GEO} is the number of GEO perturbations. If that value of k is higher than the maximum allowed value, then choose k to be the maximum allowed value instead.

The .KN-RULE keyword section may be omitted entirely and the value of k will then be set equal to zero by default. We stress that for many properties, this default choice may lead to dramatically increased calculation times compared to the best choice for k that could have been made.

4.9.7 *RESPONSE FUNCTION example

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**. Please read Section 4.9.1 before proceeding.

This section shows OPENRSP input section examples for the calculation of the response properties associated with the following molecular properties:

- Cubic force constants, i.e. the third geometric derivative of the energy $\frac{\partial E}{\partial X_i \partial X_j \partial X_k}$ where X_i, \dots are Cartesian coordinates of the system:

```

**OPENRSP
*RESPONSE FUNCTION
$PROPERTY
.PERTURBATION
3
GEO
GEO
GEO
.KN-RULE
1

```

We note that all geometric perturbations will have their associated frequency set equal to zero and there is therefore no need for a .FREQUENCY section.

- A combined run for the tensor associated with the polarizability gradient $\frac{\partial \alpha_{\alpha\beta}(-\omega; \omega)}{\partial X_i}$, i.e. the first geometric derivative of the second-order electric-dipole-perturbed energy, for the frequency configuration $\omega = 0.072$, and the tensor associated with the first hyperpolarizability Hessian $\frac{\partial^2 \beta_{\alpha\beta\gamma}(-(\omega_1 + \omega_2); \omega_1, \omega_2)}{\partial X_i \partial X_j}$, i.e. the second geometric derivative of the third-order electric-dipole-perturbed energy, for the two frequency configurations $(\omega_1, \omega_2) = \{(-0.072, 0.072), (0.036, 0.036)\}$:

```

**OPENRSP
*RESPONSE FUNCTION
$PROPERTY
.PERTURBATION
3
GEO
EL
EL
.FREQUENCY
1
0.072
.KN-RULE
0
$PROPERTY

```

```
.PERTURBATION
5
GEO
GEO
EL
EL
EL
.FREQUENCY
2
-0.072
0.072
0.036
0.036
.KN-RULE
1
```

We note that e.g for the tensor associated with the polarizability gradient, since $\alpha_{\alpha\beta}(-\omega; \omega) = -\langle\langle \hat{\mu}_\alpha; \hat{\mu}_\beta \rangle\rangle_\omega$, the elements of the calculated response tensor will be the negatives of the corresponding elements of $\frac{\partial \alpha_{\alpha\beta}(-\omega; \omega)}{\partial X_i}$.

4.9.8 Format of returned data from OpenRSP

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**. Please read Section 4.9.1 before proceeding.

The final results from OPENRSP will be returned as the file `rsp_tensor` which has the following general structure:

```
VERSION
<Version number (currently 1)>
NUM_PROPERTIES
<Number of different properties in this file>
NEW_PROPERTY [Header to mark start of data for a new property]
ORDER
<Order of energy derivative for this property>
NUM_FREQ_CFGS
<Number of different frequency configurations for this property>
OPERATORS
<Perturbing operator 1>
<Perturbing operator 2>
```

```

...
NUM COMPONENTS
<Number of components for perturbation 1>
<Number of components for perturbation 2>
...
FREQUENCIES [Header to mark start of frequency information]
CONFIGURATION [Header to mark start of frequency configuration]
<Frequency corresponding to perturbation 1
<Frequency corresponding to perturbation 2
...
CONFIGURATION [Header to mark start of next frequency configuration (if applicable)]
<Frequency corresponding to perturbation 1
<Frequency corresponding to perturbation 2
...
VALUES [Header to mark start of response tensor values]
CONFIGURATION [Start of values for frequency configuration #1]
          1          1 [Tensor element (here: the ``11'' element)]
<Value of tensor element specified above>
          1          2 [Tensor element (here: the ``12'' element)]
<Value of tensor element specified above>
...
CONFIGURATION [Start of values for frequency configuration #2 (if applicable)]
...
NEW_PROPERTY [Header to mark start of data for a new property (if applicable)]
...

```

Please note the following about the information in the `rsp_tensor` file:

- If the `$WRITE_THRES` optional keyword (described in Section 4.9.9.2) was used, elements with absolute value below the threshold thus given will not be printed to `rsp_tensor`.
- For tensor values that are equal by symmetry induced by equivalence between two or more perturbations — equivalence in this case meaning that both the perturbing operators and the associated frequencies are equal — the output file will only output a value for the first (i.e. lowest-numbered) tensor index associated with such a collection of elements. For example, for the case of the cubic force constants in the example shown in Section 4.9.7, the 123, 132, 213, 231, 312, and 321 elements are all equal by such symmetry, and thus, out of these elements, only the 123 element will be printed (unless the `$WRITE_THRES` optional keyword was used and the absolute value of this

element happens to be smaller than the threshold thus specified, in which case none of the elements listed here would be printed).

- The perturbations in the tuple making up the perturbations with respect to which a response tensor is calculated will be sorted by OPENRSP and the indices preceding each printed tensor element value follow this sorted ordering. Please note that this order may differ from the one used in a given input file.
 - The ordering can be described as follows:
 1. A perturbation of greater dimension will be listed before a perturbation of lower dimension.
 2. If two perturbations have equal dimension but have differing perturbation labels, they will be sorted in reverse alphabetical order.
 3. If two perturbations are equivalent in the two preceding respects, they will be sorted in increasing order of frequency value (negative to positive).
 - The ordering of perturbations used for each tensor printed in `rsp_tensor` can be found as follows:
 - * The tuple of perturbation labels is ordered according to the order of printing in the appropriate OPERATORS section.
 - * For each frequency configuration for a given tuple of perturbations, the associated frequencies of the perturbations are ordered according to the order of printing in the appropriate CONFIGURATION subsection of the FREQUENCIES section.

4.9.9 Other optional OpenRSP input

IMPORTANT: The version of OPENRSP included in this release of LSDALTON is a **limited version**. Please read Section 4.9.1 before proceeding.

It is possible to specify one or more pieces of optional input for the calculation. This input, if it is to be given, must be entered directly following the `*RESPONSE FUNCTION` keyword and directly before the first instance of the `$PROPERTY` keyword.

4.9.9.1 \$RESTART_FLAG

NOTE: The restarting and intermediate result storage functionality described here may be removed in future versions and should be regarded as particularly experimental.

Specify the use or non-use of intermediate result storage and retrieval. Default: Non-use. Usage:

`$RESTART_FLAG`
<Integer flag: 0 or 3>

OPENRSP may be set up to store intermediate results obtained during the calculation in cache files, and to retrieve these results and resume the calculation from the nearest possible point if the execution was interrupted (as e.g. may happen if the calculation runs out of time on a batch queueing system).

The `$RESTART_FLAG` keyword specifies if this restarting and intermediate result storage functionality is to be used. There are two valid values:

0 (default): Do not use any existing restarting information and do not save any intermediate results. If the `$RESTART_FLAG` keyword is not used, this will be the default option.

3: Save intermediate results and use any existing such stored results if available. Please note the following requirements and limitations when this option is used:

- The intermediate results will be written to files in the directory from which the execution of LSDALTON was started. The names of these files will begin with one of `OPENRSP` or `qcmatrix`.
- The number of such cache files may grow quite large and the extent of hard drive I/O associated with this storage (and retrieval) may have a noticeable impact on the calculation time and other concurrent hard drive I/O taking place on the computer system.
- For the purpose of retrieving such stored information, OPENRSP will only attempt to find such information in the root directory from which LSDALTON was started. For restarting an interrupted execution, the user must therefore take care to put all the cache files (i.e. all the files with names beginning with `OPENRSP` or `qcmatrix`) created in a previous execution into this directory. On a batch queueing system using a separate “scratch” space for calculation jobs, this may entail having a run script copy these files into the directory from which LSDALTON is to be run.
- OPENRSP makes no attempt to verify that any information retrieved from the cache files is correct in the context of the present execution. The user must make sure that the setup for the calculation for which resumption is desired is identical to the one for which the cache files was created.
- In a typical execution involving OPENRSP, there will be many calls to routines invoked by but external to OPENRSP. The majority of the computation time will typically take place in these external routines and a few of these calls will frequently

be the ones in which the most computation time is spent. Any calculation progress made inside such external routines is not available to store for OPENRSP until their execution is finished and the results are returned to OPENRSP. If the execution was interrupted during such an external call, any progress made during this call will therefore not be possible to later retrieve from the cache that OPENRSP creates, and the loss that this entails may possibly constitute a large part of the spent computation time in the execution that was interrupted.

4.9.9.2 WRITE_THRES

Specify a threshold value for the printing of response tensor elements. Default: No threshold. Usage:

```
$WRITE_THRES  
<Floating point number: Tensor element write threshold>
```

The file `rsp_tensor` containing the final results from OPENRSP was described in Section 4.9.8. If desired, the printing of elements of the resulting response tensor or tensors may be suppressed if the value of such elements is lower than a threshold as specified with the `$WRITE_THRES` keyword. Any elements whose value is greater than or equal to this threshold will still be printed. If the `$WRITE_THRES` keyword is not used, then the default option of no threshold will be applied and there will therefore be no suppression of printing of tensor elements based on their value.

4.9.10 OpenRSP test cases

Disclaimer: As described in Section 4.9.1, in the present version, the occurrence of passing tests of the OPENRSP functionality **does not guarantee** that the corresponding calculation setup and results are without errors.

The test cases can be found in the `LSDALTON/test/openrsp` directory, where `LSDALTON` is the root folder of the LSDALTON repository. To run test cases after building the code, please use

```
ctest
```

with any desired flags, from the directory where LSDALTON was built, noting that the OPENRSP test cases can be recognized by the `openrsp` name in the title of each. Please also note that the test case files contain input files for several properties which may serve as useful references.

4.10 **DEC and **CC

The DEC functionality is **not enabled by default** when installing LSDALTON, but can be manually activated by adding the `-DENABLE_DEC=ON` flag during setup.

Correlated coupled-cluster (CC) calculations using the Divide-Expand-Consolidate (DEC) scheme [78, 79, 80, 81, 82, 83, 84], where local orbitals are used to carry out the correlated calculation on a large molecular system in terms of many small local calculations. Formally, the computational time for the DEC scheme scales linearly with system size. DEC calculations are invoked using the `**DEC` keyword.

Furthermore, conventional coupled-cluster energies for the RI-MP2, MP2, CC2, CCSD, and CCSD(T) models are available using the `**CC` keyword. The computational time for these implementations scale with the system size to the fifth, fifth, fifth, sixth, and seventh power for RI-MP2, MP2, CC2, CCSD, and CCSD(T), respectively.

The `**DEC` and `**CC` implementations uses minimal I/O and employ massive parallelism and will therefore only be effective if many computing nodes are available.

Special note: It is mandatory that the memory available for the calculation is specified in the input, see `.MEMORY` keyword below.

The `**DEC` and `**CC` sections use local orbitals by default, although the orbitals are converted to the (pseudo-)canonical basis when the MP2 and CC equations are solved (as well as for CCSD(T)). It is encouraged to use the keyword `.CANONICAL` when using `**CC`.

Below are keywords for the `**DEC` section and `**CC` section and relevant test cases are given.

4.10.1 **DEC input keywords

`.MP2`

DEC-MP2 model requested

`.RIMP2`

DEC-RI-MP2 model requested

`.CC2`

DEC-CC2 model requested

`.CCSD`

DEC-CCSD model requested

.CCSD(T)

DEC-CCSD(T) model requested

.FOT

<Threshold>

Fragment optimization threshold (FOT), it defines the precision of a DEC calculation compared to a conventional calculation [78, 79, 80, 81, 82, 83]. Default value is 10^{-4} .a.u. The error in the total correlation energy is proportional to the FOT and the size of the system.

.DECPRINT

<Print level>

Level of information printed to LSDALTON.OUT. The input is a integer and the default value is 0.

.MEMORY

<Memory in gigabytes>

It is mandatory to specify the memory (in gigabytes) available for the calculation using this keyword! For an MPI run this is the memory available for each MPI process. If this keyword is not set, the program will stop with a quit statement informing about an error in the LSDALTON.INP file. (This keyword must also be used in ****CC** section).

.NOTABSORBH

By default orbitals originally assigned to hydrogen atoms are reassigned to the nearest heavy atom. This reassigning can be turned off by invoking this keyword.

.DENSITY

Calculate DEC-MP2 density matrix [82] (stored in MP2.dens). The MP2 electric dipole moment is also calculated if this keyword is invoked.

.GRADIENT

Calculate MP2 molecular gradient [82] at input geometry (for geometry optimization, see test case examples below). The MP2 density and electric dipole moment is also calculated if this keyword is invoked.

.KAPPATHR

<THR>

Threshold for $\bar{\kappa}$ multiplier equations solved when DEC-MP2 molecular gradient is calculation. Default value is 10^{-4} .

.MPIGROUPSIZE

<Size>

Size of local MPI groups (integer) in massively parallel DEC scheme [83]. The default settings should suffice for most purposes, and use of the keyword is only recommended for advanced users.

4.10.1.1 *LOFEX

This section can be used to calculate CC singlet excitation energies and associated oscillator strengths at a reduced computational cost using the LoFEx procedure as described in [85, 86, 87]. It needs to be combined with the **.CCRESPONSE** section as described in Section 4.10.3.1. Excitation energies can be calculated at the RI-CC2 and CCSD levels while oscillator strengths are only available at the RI-CC2 level.

.THRES

<Main LoFEx threshold in a.u.>

Threshold used to control the convergence of the XOS based on the change in the excitation energy or based on the change in oscillator strength when the *spectrum* strategy is used via the **.FORCE_SPECTRUM** keyword (see [87]). Default value is $7.35 \cdot 10^{-4}$ a.u., i.e. ~ 0.02 eV.

.NAVAT

<Number of AVerage AToms>

Defines the number of orbitals to add to the XOS after each (non-converged) LoFEx iteration. The default value is 10 which means that ten times the average number of molecular orbitals per atom are added ($10 \times N_{\text{MOs}}/N_{\text{atoms}}$).

.NTO_CCS

Use CCS/CIS natural transition orbitals (NTOs) to build the mixed orbital space instead of time-dependent Hartree–Fock NTOs which require the proper input in the ****RESPONS** section. See test cases below.

.FORCE_SPECTRUM

Use the *spectrum* strategy instead of the *standard* strategy to optimize the XOS, as described in [87]. It calculates both oscillator strength and excitation energies and requires the corresponding keywords in the **.CCRESPONSE** section.

.XOS_OPT_MODEL

<CC MODEL>

Select CC model to use for the XOS optimization. Default is RI-CC2, but CCSD can be enforced using **.CCSD** [86].

.NTO_THR**<NTO threshold>**

Use to select the CIS or TDHF NTOs to include in the mixed orbital space based on their eigenvalues. Default is $\tau_{\text{NTO}} = 0.1$ such that the sum of the NTO eigenvalues (starting from the largest eigenvalues) included in the mixed orbital space must add up to $1 - \tau_{\text{NTO}}$.

4.10.1.2 *CORNFLEX

This section can be used to calculate CC singlet excitation energies and associated oscillator strengths at a reduced computational cost using the CornFLEX procedure as described in Ref. [88]. It needs to be combined with the **.CCRESPONSE** section as described in Section 4.10.3.1. Excitation energies can be calculated at the RI-CC2 and CCSD levels while oscillator strengths are only available at the RI-CC2 level.

.NTO_THR**<CIS(D')-NTO threshold>**

Set the main CorNFLEX threshold $\tau_{\text{CIS(D)'}}$, which defines the precision of the calculation by determining how many CIS(D')-NTOs to include in the excitation orbital space (XOS) as described in Ref. [88]. The value corresponding to the standard threshold of Ref. [88] is $\tau_{\text{CIS(D)'}} = 10^{-4}$. If $\tau_{\text{CIS(D)'}}$ is zero or negative the XOS includes the full orbital space and a standard CC calculation is reproduced.

.CCS_NTO_THR**<CIS-NTO threshold>**

Set the τ_{CIS} threshold described in Ref. [88]. The default value is $\tau_{\text{CIS}} = 10^{-3}$, which was used in Ref. [88]. It is generally not recommended to change this threshold for application runs.

.FD_THR**<Diagonal Fock matrix threshold>**

Set the τ_F threshold (input in a.u.) described in Ref. [88]. The default value is $\tau_F = 1.83747$ a.u., corresponding to 50 eV, which is the value used in Ref. [88]. It is generally not recommended to change this threshold for application runs.

Example test cases for **DEC calculations:

These test cases illustrate the main features of the DEC scheme. The test cases are located in the test/dectests directory.

- *decmp2_energy*: Calculate DEC-MP2 energy.
- *decmp2_density*: Calculate DEC-MP2 density matrix and electric dipole moment.
- *decmp2_gradient*: Calculate DEC-MP2 molecular gradient.
- *decmp2_geoopt*: Carry out DEC-MP2 geometry optimization. Note that the geometry optimization is invoked by the ****OPTIMIZE** section, while the ****DEC** section specifies that the geometry optimization should use DEC-MP2 energies and molecular gradients. The DEC-MP2 geometry optimization will automatically estimate the intrinsic energy error of the DEC calculation. If this error is larger than the difference between two geometry, the FOT level will be increased to get a more accurate energy and gradient in the next iteration.
- *loflex_ccs_ricc2*: Calculate RI-CC2 excitation energies and oscillator strength using LoFEx and CCS-NTOs.
- *loflex_tdhf_ccsd*: Calculate CCSD excitation energies using LoFEx and TDHF-NTOs. The XOS optimization is also enforced at the CCSD level which is not default.
- *cornflex_ricc2*: Calculate RI-CC2 excitation energies using the CorNFLEEx procedure.
- *cornflex_ricc2_spectrum*: Calculate RI-CC2 excitation energies and oscillator strengths using the CorNFLEEx procedure.

4.10.2 ****CC input keywords**

.CCS

Use CCS model for excited states (see Section [4.10.3.1](#)).

.RICCS

Use CCS model and the density fitting approximation for the Fock matrix builds.

.MP2

Use MP2 model. This gives the canonical MP2 energy and is not optimized for parallel performance. It is only intended to be used for benchmarking DEC-MP2 calculations on relatively small molecular systems.

.RIMP2

Use RI-MP2 model. This gives the canonical RI-MP2 energy and is optimized for parallel performance. It is encourage to use with the **.CANONICAL** keyword

.CC2

Use CC2 model which is not an optimized code, but rather uses the CCSD optimized code with some contributions removed. See CCSD documentation for details.

.RICC2

Use CC2 model and the RI approximation for the calculation of two-electron integrals. The implementation is based on Refs. [89, 90]. Three-center integrals are kept in core memory (not disk), the implementation is OMP parallelized (but not MPI!).

.CCSD

Use CCSD model. The code employed is a massively parallel version and flexible with respect to memory. Due to the use of one-sided communication in these models it is recommended to run CCSD with an asynchronous progress engine switched on (how to do that can be found in the documentation of the respective MPI library you are using). When running a CCSD calculation make sure that at least one quantity of size V^2O^2 (O = number of occupied orbitals, V = number of virtual orbitals) fits into memory in double precision floating point numbers (with some additional space of course). Using this keyword automatically triggers checkpointing after each iteration. The calculation can be restarted with the usual **.RESTART** keyword.

.CCSD(T)

Use the CCSD(T) model. This gives the canonical CCSD(T) energy and is optimized for parallel performance. The **.CANONICAL** keyword must be used for CCSD(T).

.MEMORY

<Memory in gigabytes>

It is mandatory to specify the memory (in gigabytes) available for the calculation using this keyword! For an MPI run this is the memory available for each MPI process. If this keyword is not set, the program will stop with a quit statement informing about an error in the LSDALTON.INP file. (This keyword must also be used in ****DEC** section).

.SNOOP

Calculate intermolecular interaction energies using the Same Number Of Optimized Parameters (SNOOP) scheme [91]. This can be done for the RIMP2, MP2, CCSD, and CCSD(T) models. It is important to specify the subsystems in the molecule input file. Example inputs are given in the "snoop" test cases in the test/dectests directory.

.SNOOPTHR

The threshold for the Hartree-Fock optimization of SNOOP monomers. Default value

is very tight (10^{-7}), and it is recommended to keep it tight due to a simplistic implementation of SCF loop.

.SNOOPMAXITER

Maximum number of iterations in SNOOP monomer Hartree–Fock calculations.

.SNOOPMAXDIIS

Maximum number of DIIS vectors to store in SNOOP HF calculations (RH/DIIS scheme). If the SNOOP monomer Hartree–Fock calculations do not converge, it might be advantageous to increase this value (default value is 5).

.SNOOPRESTART

Restart SNOOP calculation. It requires SNOOP restart files (SNOOP*****.restart).

Example test cases for **CC calculations:

The test cases are located in LSDALTON/test/dectests.

- *fullmp2_energy*: Calculate conventional canonical MP2 energy.
- *fullccsd_high*: Calculate conventional canonical CCSD energy.
- *ricc2_spectrum*: Calculate CC2 oscillator strengths (2 singlet states in length gauge).

4.10.3 Common **DEC and **CC input keywords

.HFRESTART

Start DEC/CC calculation from HF restart files *dens.restart*, *fock.restart*, *lcm_orbitals.u*, and *overlapmatrix*

.RESTART

Restart unfinished DEC/CC calculation. Requires that HF files *dens.restart*, *fock.restart*, *lcm_orbitals.u*, and *overlapmatrix* are present in the folder where the calculation is restarted. Additionally, if .info files are present it means that some of the fragment calculations have been carried out and the .RESTART keyword will read information about the finished fragment calculations from file and calculate the missing fragments. Specifically, if the files *atomicfragments.info* and *atomicfragmentsdone.info* are present, it means that some (or all) of the atomic fragment calculations are finished. If, in addition, the file *fragenergies.info* (energy calculation) *or* *mp2grad.info*

(MP2 density or gradient calculation) is also present, it means that some (or all) of the pair fragments are also done.

Note 1: This keyword should NOT be combined with `.RESTART` in the `*DENSOPT` section.

Note 2: This keyword cannot be used for a DEC-MP2 geometry optimization. The only way to "restart" a DEC-MP2 geometry optimization is use the current geometry in a new DEC-MP2 geometry optimization – i.e., copy the most recently generated `MOLECULE.xxx` file to `MOLECULE.INP` and run the calculation again using the same `LSDALTON.INP`.

`.FROZENCORE`

Use the frozen core approximation.

`.CANONICAL`

Use canonical orbitals. This is recommended in combination with `**CC` while it is only recommended for advanced users for testing purposes in connection with the `**DEC` keyword. In general, the DEC scheme is only meaningful for local orbitals.

`.CCMAXITER`

Maximum number of iterations in CC solver.

`.CCTHR`

Convergence threshold for residual norm in CC solver.

`.SUBSIZE`

Number of previous vectors to store in CROP scheme [50] for CC solver.

`.CCSDNOSAFE`

Prevent saving of CCSD amplitudes as checkpoint files.

`.INTEGRALTHRESHOLD`

Integral screening threshold in the correlation treatment.

4.10.3.1 `*CCRESPONSE`

This section can be used to calculate CC singlet excitation energies and associated oscillator strengths.

`.NEXCIT`

Number of excitation energies to be calculated (available for the CCS, RICCS, CC2, RICC2 and CCSD models).

.SPECTRUM

Calculate linear response CC oscillator strengths (in the length gauge by default) [92]. Only available at the CCS, RICCS, and RICC2 levels.

.OPERATOR

Operator string

Select the operator used to calculate oscillator strengths: DIPLN for the length gauge or DIPVEL for the velocity gauge.

.THRES

Threshold

Threshold used in the Davidson solver. An excitation energy is converged when the norm of the corresponding residual vector is below the **Threshold**, the default value is $1.0e-4$ a.u.

.MACRO_THR

Threshold

Excitation energy threshold used to check for self-consistency in the non-linear Davidson solver. Default value is $1.0e-4$ a.u. (Only used for RICC2 model) [89, 93].

4.11 **PLT

Generation of .plt files which may be used to visualize densities, orbitals, and electrostatic potentials by calculating these at specific points in space. The .plt files can be visualized e.g. using the Chimera program [1].

Important note: This section assumes that an LSDALTON calculation has already been carried out to generate files containing density matrix elements or molecular orbital coefficients. The .plt files can then be generated by running a new calculation where the LSDALTON.INP file has been modified as exemplified below.

The density matrix from HF and DFT calculation are saved in a file *dens.restart*, while the MP2 density is saved in the file *MP2.dens* (if requested, see **DEC section). The canonical MO coefficients are saved in the file *cmo_orbitals.u*, while localized molecular orbitals are saved in a file *lcm_orbitals.u* (if requested). The definition of the grid points is described in the **PLTGRID section.

Note that it is also possible to generate .plt files for orbitals on the run as described in Section 4.7.

.INPUT

<Name of input file>

The name of the input file containing density matrix elements or orbital coefficients. Possible input files include *dens.restart*, *MP2.dens*, *cmo_orbitals.u*, and *lcm_orbitals.u*.

.OUTPUT

<Name of output file>

The name of the output plt-file where the calculated values at each grid point are saved. Examples are given below.

.DENS

Construct .plt file for electron density at grid points. For example to calculate electron density from the file *dens.restart* and save information in file *dens.plt*, insert the following section in the LSDALTON.INP file used to generate the *dens.restart* file in the first place:

```
**PLT
.INPUT
dens.restart
.OUTPUT
dens.plt
.DENS
```

.EP

Construct .plt file for electrostatic potential grid points based on input density matrix. For example to calculate electrostatic potential from the density in the file *dens.restart* and save information in file *ep.plt*, insert the following section in the LSDALTON.INP file used to generate the *dens.restart* file in the first place:

```
**PLT
.INPUT
dens.restart
.OUTPUT
ep.plt
.EP
```

.ORB

<Orbital index>

Construct .plt file for specific orbital at grid points. For example to calculate values of orbital number 8 at grid points from the file *lcm_orbitals.u* and save information in file *orb8.plt*, insert the following section in the LSDALTON.INP file used to generate the *lcm_orbitals.u* file in the first place:


```

**PLT
.INPUT
lcm_orbitals.u
.OUTPUT
orb8.plt
.ORB
8

```

.CHARGEDIST

```
<Orbital index 1      Orbital index 2>
```

Construct .plt file for charge distribution of two orbitals at grid points. For example to calculate charge distribution between orbitals 5 and 8 at grid points from the file *lcm_orbitals.u* and save information in file *cd_5_8.plt*, insert the following section in the LSDALTON.INP file used to generate the *lcm_orbitals.u* file in the first place::

```

**PLT
.INPUT
lcm_orbitals.u
.OUTPUT
cd_5_8.plt
.CHARGEDIST
5 8

```

4.12 **PLTGRID

Definition of 3-dimensinal grid used for generation of .plt files in **PLT section. The default grid choice should suffice for most visualization purposes. This section is only for the advanced user who wishes to modify the default choice of grid. This also modifies the grid used when .plt files for local orbitals are generated on the run, see Section 4.7.

The grid box is defined in the following manner:

- The first point in the grid box is (X1,Y1,Z1).
- The remaining grid points are then defined by going out in the X, Y, and Z directions with step sizes *deltax*, *deltay*, and *deltaz*, until there are *nX*, *nY*, and *nZ* points in the X,Y, and Z directions (giving a total number of gridpoints $nX \times nY \times nZ$).

There are two options to define the grid box: (i) manual gridbox where X1, Y1, Z1, *deltax*, *deltay*, *deltaz*, *nX*, *nY*, *nZ* are defined explicitly in the input file; (ii) molecule-specific gridbox where all atoms are contained within the grid box AND there is an additional buffer zone around the outermost atoms in the X, Y, and Z directions. Option (ii) thus requires

that the `deltax`, `deltay`, `deltaz`, and `buffer` values are defined by the input, which implicitly defines `X1`, `Y1`, `Z1`, `nX`, `nY`, and `nZ` based on the molecular structure.

The default gridbox uses option (ii) with `deltax=deltay=deltaz=0.3` a.u. and `buffer=6.0` a.u. User-defined grid box based on options (i) and (ii) use the following inputs:

`.MANUAL`

```
<X1  Y1  Z1>
<deltax  deltay  deltaz>
<nX  nY  nZ>
```

Manual definition of gridbox as described above. All input values are in a.u.

`.MOLECULE`

```
<buffer>
<deltax  deltay  deltaz>
```

Molecule-specific definition of gridbox as described above. All input values are in a.u.

4.13 ****PCM**

Keywords specified in this section set up the inclusion of a continuum description of the solvent using the polarizable continuum model for solvation [2, 3]. In the current implementation, it is possible to use a PCM description of the solvent for SCF wave functions and for electric linear response properties only. LSDALTON uses the external `PCMSolver` library [7], freely available under the terms of the LGPLv3 license. Setting up a PCM calculation requires that the user provides an additional input file to be used by the external library, refer to <http://pcmsolver.readthedocs.org/en/latest/users/input.html> Examples of `PCMSolver` input files can be found in `test/pcm/energy/pcmsolver.inp`, `test/pcm/response/eq_pcmsolver.inp` and `test/pcm/energy/noneq_pcmsolver.inp`

The `lsdalton` launch script accepts the third input file as a third positional argument. For example:

```
lsdalton lsdalton.inp molecule.mol pcmsolver.inp
```

will run the PCM calculation and produce an output file with name `lsdalton_molecule_pcmsolver.out`

`.SEPARATE`

Triggers whether the nuclear and electronic electrostatic potential and apparent surface charge should be treated separately. This is a debug option.

4.14 ****WANNIER**

NB: THE WANNIER CODE IS NOT YET WORKING.

Do SCF calculations for translation invariant systems using the Wannier - Roothaan - Hall algorithm.

...

.AO_CUTOFF

<NX NY NZ>

Determines the domain \mathcal{D}_{AO} as the number of layers of cells around the reference cell along the x, y and z direction. The molecular coefficients are expanded only in the atomic orbitals in this domain.

The default value is set to [2 2 2] .

.ORTHPOT_STRENGTH

lambda The strength of the orthogonalizing potential λ .

The default value is set to 10^4 .

.NF_CUTOFF

<NX NY NZ>

The nearfield cutoff for the Coloumb interaction. Specified as a number of layers along the x, y and z axis around the reference cell. The contribution to the Fock matrix is computed exactly in this domain.

The default value is set to [5 5 5] .

.SCREENING_THRESHOLD

Integral screening threshold.

The default value is set to 10^{-14} .

.NO_REDUNDANT

Hermiticity is assumed and only the positive cells are computed. By default all cells are computed

.DEBUG_MODE

N

Different level of testing and printing for the Wannier code

- N < 0 No test are run, to be used with extreme caution
- N = 0 [Default] Sanity tests are run - No debug printing
- N = 1 + print the MO coefficients after the Löwdin orthogonalisation
- N = 2 + print the density and overlap matrices
- N = 3 + print the MO coefficients before the Löwdin orthogonalisation

- N = 4 + print intermediate steps in Löwdin
- N = 5 + print the Fock matrix and the orthogonalizing potential
- N = 6 + print the Fock matrix components
- N > 7 + print whatever still remains to be printed!

.CONV_STARTGUESS

The PBC calculation starts from the converged molecular orbitals of the reference cell taken as a molecule

.AO_SHUKLA

(Default) Turns on the atomic-orbital formulation of Shukla's algorithm [94] (default).

Warning: The orthogonalizing potential is built on the occupied orbitals only.

.MO_SHUKLA

Turns on the molecular-orbital formulation of Shukla's algorithm [94].

Bibliography

- [1] E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D.M. Greenblatt, E. C. Meng, and T. E. Ferrin. *J. Comp. Chem.*, 25:1605, 2004.
- [2] Jacopo Tomasi, Benedetta Mennucci, and Roberto Cammi. Quantum Mechanical Continuum Solvation Models. *Chem. Rev.*, 105(8):2999–3094, 2005.
- [3] Monica Bugeanu, Roberto Di Remigio, Krzysztof Mozgawa, Simen Sommerfelt Reine, Helmut Harbrecht, and Luca Frediani. Wavelet Formulation of the Polarizable Continuum Model. II. Use of Piecewise Bilinear Boundary Elements. *Phys. Chem. Chem. Phys.*, 2015.
- [4] B. Jansík, S. Høst, M. P. Johansson, J. Olsen, and P. Jørgensen. Robust and Reliable Multilevel Minimization of the Kohn—Sham Energy. *J. Chem. Theory Comput.*, 5:1027, 2009.
- [5] B. Jansík, S. Høst, M. P. Johansson, J. Olsen, and P. Jørgensen. A stepwise atomic, valence—molecular, and full—molecular optimisation of the Hartree—Fock/Kohn—Sham energy. *Phys. Chem. Chem. Phys.*, 11:5805, 2009.
- [6] T. Helgaker, P. Jørgensen, and J. Olsen. *Molecular Electronic Structure Theory, First Edition*. Wiley, 2000.
- [7] PCMSolver, an Application Programming Interface for the Polarizable Continuum Model electrostatic problem, written by R. Di Remigio, L. Frediani and K. Mozgawa, (see <http://pcmsolver.readthedocs.org>).
- [8] Manuel Guidon, Jürg Hutter, and Joost Vandevondele. Auxiliary density matrix methods for Hartree-Fock exchange calculations. *Journal of Chemical Theory and Computation*, 6(8):2348–2364, August 2010.
- [9] Patrick Merlot, Róbert Izsák, Alex Borgoo, Thomas Kjærgaard, Trygve Helgaker, and Simen Reine. Charge-constrained auxiliary-density-matrix methods for the hartree-fock exchange contribution. *Journal of Chemical Physics*, 141:094104, August 2014.

- [10] Yihan Shao and Martin Head-Gordon. An improved j matrix engine for density functional theory calculations. *Chem. Phys. Lett.*, 323(5-6):425, 2000.
- [11] Yihan Shao, Christopher A. White, and Martin Head-Gordon. Efficient evaluation of the coulomb force in density-functional theory calculations. *J. Chem. Phys.*, 114(15):6572, 2001.
- [12] Christian Ochsenfeld, Christopher A. White, and Martin Head-Gordon. Linear and sublinear scaling formation of hartree-fock-type exchange matrices. *J. Chem. Phys.*, 109(5):1663, 1998.
- [13] Patrick Merlot, Thomas Kjaergaard, Trygve Helgaker, Roland Lindh, Francesco Aquilante, Simen Reine, and Thomas Bondo Pedersen. Attractive electron-electron interactions within robust local fitting approximations. *J. Comp. Chem.*, pages 1–11, April 2013.
- [14] L E McMurchie and E R Davidson. One- and two-electron integrals over cartesian gaussian functions. *J. Comp. Phys.*, 26:218–231, 1978.
- [15] Simen Reine, Erik Tellgren, and Trygve Helgaker. A unified scheme for the calculation of differentiated and undifferentiated molecular integrals over solid-harmonic gaussians. *Phys. Chem. Chem. Phys.*, 9:4771–4779, 2007.
- [16] P. Hohenberg and W. Kohn. *Phys. Rev.*, 136:B864, 1964.
- [17] W. Kohn and L. J. Sham. *Phys. Rev.*, 140:A1133, 1965.
- [18] J. C. Slater. *Quantum Theory of Molecular and Solids*, volume 4, chapter The Self-Consistent Field for Molecular and Solids. McGraw-Hill, New York, 1974.
- [19] A. D. Becke. *Phys. Rev. A*, 38:3098, 1988.
- [20] N. C. Handy and A. J Cohen. *Mol. Phys.*, 99:403, 2001.
- [21] J. P. Perdew, K. Burke, and M. Ernzerhof. *Phys. Rev. Lett.*, 77(3865), 1996.
- [22] J. P. Perdew and Y. Wang. *Phys. Rev. B*, 33:8800, 1986.
- [23] S. H. Vosko, L. Wilk, and M. Nusair. *Can. J. Phys.*, 58:1200, 1980.
- [24] C. Lee, W. Yang, and R. G. Parr. *Phys. Rev. B*, 57:785, 1988.
- [25] B. Miehlich, A. Savin, H. Stoll, and H. Preuss. *Chem. Phys. Lett.*, 157:200, 1989.
- [26] J. P. Perdew. *Phys. Rev. B*, 33:8822, 1986.

- [27] J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, and C. Fiolhais. *Phys. Rev. B*, 46:6671, 1992.
- [28] J. P. Perdew and A. Zunger. *Phys. Rev. B*, 23:5048, 1981.
- [29] R. van Leeuwen and E. J. Baerends. Exchange-correlation potential with correct asymptotic behavior. *Phys. Rev. A*, 1994.
- [30] A. D. Becke. *J. Chem. Phys.*, 98:5648, 1993.
- [31] T. Yanai, D. P. Tew, and N. C. Handy. A new hybrid exchange-correlation functional using the Coulomb-attenuating method (cam-b3lyp). *Chem. Phys. Lett.*, 393:51, 2004.
- [32] T. W. Keal and D. J. Tozer. The exchange-correlation potential in kohn-sham nuclear magnetic resonance shielding calculations. *J. Chem. Phys.*, 119:3015, 2003.
- [33] T. W. Keal D. J. Tozer and T. Helgaker. Giau shielding constants and indirect spin-spin coupling constants: performance of density functional methods. *Chem. Phys. Lett.*, 391:374, 2004.
- [34] T. W. Keal and D. J. Tozer. A semi-empirical generalised gradient approximation exchange-correlation functional. *J. Chem. Phys.*, 121:5654, 2004.
- [35] C. Adamo and V. Barone. *J. Chem. Phys.*, 110:6158, 1999.
- [36] Ulf Ekström, Lucas Visscher, Radovan Bast, and Andreas J Thorvaldsen. Arbitrary-Order Density Functional Response Theory. *J. Chem. Theory Comput.*, 6:1971–1980, 2010.
- [37] P. Salek and A. Hesselmann. A self-contained and portable density functional theory library for use in Ab Initio quantum chemistry programs. *J. Comput. Chem.*, 28:2569, 2007.
- [38] N. C. Handy C. W. Murray and G. J. Laming. *Mol. Phys.*, 78:997, 1993.
- [39] S. Grimme. *J. Comp. Chem.*, 27:1787, 2006.
- [40] S. Grimme, J. Antony, S. Ehrlich, and H. Krieg. *J. Chem. Phys.*, 132:154104, 2010.
- [41] S. Grimme, S. Ehrlich, and L. Goerigk. *J. Comp. Chem.*, 32:1456, 2011.
- [42] P.-A. Malmqvist R. Lindh and L. Gagliardi. Molecular integrals by numerical quadrature. i. radial integration. *Theor. Chem. Acc.*, 106:178–187, 2001.
- [43] O. Treutler and R. Ahlrichs. *J. Chem. Phys.*, 102:346, 1995.

- [44] A. D. Becke. *J. Chem. Phys.*, 88:2547, 1988.
- [45] G. Scuseria R. E. Stratmann and M. J. Frisch. *Chem. Phys. Lett.*, 257:213, 1996.
- [46] P. Macak M. A. Watson, P. Salek and T. Helgaker. *J. Chem. Phys.*, 121:2915, 2004.
- [47] S. Hirata, C.G. Zhan, E. Aprà, T.L. Windus, and D.A.Dixon. A new, self-contained asymptotic correction scheme to exchange-correlation potentials for time-dependent density functional theory. *J. Phys. Chem. A*, 107:10154, 2003.
- [48] M.E. Casida and D.R. Salahub. Asymptotic correction approach to improving approximate exchange–correlation potentials: Time-dependent density-functional theory calculations of molecular excitation spectra. *J. Chem. Phys.*, 113:8918, 2000.
- [49] C.G.Zhan, J.A. Nichols, and D.A. Dixon. Ionization potential, electron affinity, electronegativity, hardness, and electron excitation energy: Molecular properties from density functional theory orbital energies. *J. Phys. Chem. A*, 107:4184, 2003.
- [50] M. Ziólkowski, V. Weijo, Poul Jørgensen, and J. Olsen. An efficient algorithm for solving nonlinear equations with a minimal number of trial vectors: Applications to atomic-orbital based coupled-cluster theory. *J. Chem. Phys.*, 128:204105, 2008.
- [51] S. Høst, B. Jansík, J. Olsen, P. Jørgensen, S. Reine, and T. Helgaker. A ground-state-directed optimization scheme for the Kohn–Sham energy. *Phys. Chem. Chem. Phys.*, 10:5344, 2008.
- [52] S. Høst, J. Olsen, B. Jansík, L. Thøgersen, Poul Jørgensen, and T. Helgaker. The augmented Roothaan–Hall method for optimizing Hartree–Fock and Kohn–Sham density matrices. *J. Chem. Phys.*, 129:124106, 2008.
- [53] Ernest R. Davidson. *J. Comput. Phys.*, 17:87, 1975.
- [54] P. Pulay. *Chem. Phys. Lett.*, 73:393, 1980.
- [55] P. Pulay. *J. Comp. Chem.*, 3:556, 1982.
- [56] J. H. Van Lenthe, R. Zwaans, H. J. J. Van Dam, and M. F. Guest. Starting SCF calculations by superposition of atomic densities. *J. Comp. Chem.*, 27:926, 2006.
- [57] S. Reine, A. Krapp, M. F. Iozzi, V. Bakken, T. Helgaker, F. Pawłowski, and P. Salek. An efficient density-functional-theory force evaluation for large molecular systems. *J. Chem. Phys.*, 133:044102, 2010.
- [58] V. Bakken and T. Helgaker. The efficient optimization of molecular geometries using redundant internal coordinates. *J. Chem. Phys.*, 117:9160, 2002.

- [59] 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.
- [60] J. Baker. Techniques for geometry optimization: A comparison of cartesian and natural internal coordinates. *J. Comp. Chem.*, 14:1085, 1993.
- [61] L. Verlet. Computer "experiments" on classical fluids. i. thermodynamics properties of lennard-jones molecules. *Phys. Rev.*, 159:98, 1967.
- [62] P. Pulay and G. Fogarasi. Fock matrix dynamics. *Chem. Phys. Lett.*, 386:272, 2004.
- [63] A. M. N. Niklasson, C. J. Tymczak, and M. Challacombe. Time-reversible born-oppenheimer molecular dynamics. *Phys. Rev. Lett.*, 97:123001, 2006.
- [64] I.-M. Høyvik, B. Jansík, and P.Jørgensen. Trust region minimization of orbital localization functions. *J. Chem. Theory Comput.*, 8:3137, 2012.
- [65] I.-M. Høyvik, B. Jansík, K. Kristensen, and P.Jørgensen. Local hartree-fock orbitals using a three-level optimization strategy for the energy. *J. Comp. Chem.*, 34:1311, 2013.
- [66] B. Jansík, S. Høst, K.Kristensen, and P.Jørgensen. Local orbitals by minimizing powers of the orbital variance. *J. Chem. Phys.*, 134:194104, 2011.
- [67] I.-M. Høyvik, B. Jansík, and P.Jørgensen. Orbital localization using fourth central moment minimization. *J. Chem. Phys.*, 137:224114, 2012.
- [68] I.-M. Høyvik, B. Jansík, and P.Jørgensen. Pipek-mezey localization of occupied and virtual orbitals. *J. Comp. Chem.*, page (DOI: 10.1002/jcc.23281), 2013.
- [69] A. J. Thorvaldsen, K. Ruud, K. Kristensen, P. Jørgensen, and S. Coriani. A density matrix-based quasienergy formulation of the kohn–sham density functional response theory using perturbation- and time-dependent basis sets. *J. Chem. Phys.*, 129:214108, 2008.
- [70] S. Coriani, S. Høst, B. Jansík, L. Thøgersen, J. Olsen, P. Jørgensen, S. Reine, F. Pawłowski, T. Helgaker, and P. Sałek. Linear-scaling implementation of molecular response theory in self-consistent field electronic-structure theory. *J. Chem. Phys.*, 126:154108, 2007.
- [71] J. Kauczor, P. Jørgensen, and P. Norman. On the efficiency of algorithms for solving hartree–fock and kohn–sham response equations. *J. Chem. Theory Comput.*, 7:1610, 2011.

- [72] Kasper Kristensen, Joanna Kauczor, Andreas J. Thorvaldsen, Poul Jørgensen, Thomas Kjærgaard, and Antonio Rizzo. *J. Chem. Phys.*, 134:214104, 2011.
- [73] T. Kjærgaard, P. Jørgensen, P. Salek A. J. Thorvaldsen, and S. Coriani. A gauge-origin independent formulation and implementation of magneto-optical activity within atomic-orbital-density based hartree-fock and kohn-sham response theories. *J. Chem. Theory Comput.*, 5:1997, 2009.
- [74] T. Kjærgaard, K. Kristensen, J. Kauczor, P. Jørgensen, S. Coriani, and A. J. Thorvaldsen. Comparison of standard and damped response formulations of magnetic circular dichroism. *J. Chem. Phys.*, 135:024112, 2011.
- [75] Richard L. Martin. Natural transition orbitals. *J. Chem. Phys.*, 118(11):4775–4777, 2003.
- [76] Thibaud Etienne. Transition matrices and orbitals from reduced density matrix theory. *J. Chem. Phys.*, 142(24):244103, jun 2015.
- [77] K. Kristensen, P. Jørgensen, A. J. Thorvaldsen, and T. Helgaker. Efficient elimination of response parameters in molecular property calculations for variational and nonvariational energies. *J. Chem. Phys.*, 129:214103, 2008.
- [78] Marcin Ziólkowski, Branislav Jansík, Thomas Kjærgaard, and Poul Jørgensen. Linear scaling coupled cluster method with correlation energy based error control. *J. Chem. Phys.*, 133:014107, 2010.
- [79] Kasper Kristensen, Marcin Ziólkowski, Branislav Jansík, Thomas Kjærgaard, and Poul Jørgensen. *J. Chem. Theory Comput.*, 7:1677, 2011.
- [80] Ida-Marie Høyvik, Kasper Kristensen, Branislav Jansík, and Poul Jørgensen. *J. Chem. Phys.*, 136:014105, 2012.
- [81] Kasper Kristensen, Ida-Marie Høyvik, Branislav Jansík, Poul Jørgensen, Thomas Kjærgaard, Simen Reine, and Jacek Jakowski. Mp2 energy and density for large molecular systems with internal error control using the divide-expand-consolidate scheme. *Phys. Chem. Chem. Phys.*, 14:15706, 2012.
- [82] Kasper Kristensen, Poul Jørgensen, Branislav Jansík, Thomas Kjærgaard, and Simen Reine. Molecular gradient for second-order møller-plesset perturbation theory using the divide-expand-consolidate (dec) scheme. *J. Chem. Phys.*, 137:114102, 2012.
- [83] Kasper Kristensen, Thomas Kjærgaard, Ida-Marie Høyvik, Patrick Ettenhuber, Poul Jørgensen, Branislav Jansik, Simen Reine, and Jacek Jakowski. The divide-expand-consolidate mp2 scheme goes massively parallel. *Mol. Phys.*, 111:1196, 2013.

- [84] Janus J. Eriksen, Pablo Baudin, Patrick Ettenhuber, Kasper Kristensen, Thomas Kjærgaard, and Poul Jørgensen. Linear-Scaling Coupled Cluster with Perturbative Triple Excitations: The Divide-Expand-Consolidate CCSD(T) Model. *J. Chem. Theory Comput.*, 11:2984, 2015.
- [85] Pablo Baudin and Kasper Kristensen. LoFEx — A local framework for calculating excitation energies : Illustrations using RI-CC2 linear response theory. *J. Chem. Phys.*, 144(22):224106, 2016.
- [86] Pablo Baudin, Dmytro Bykov, Dmitry Liakh, Patrick Ettenhuber, and Kasper Kristensen. A local framework for calculating coupled cluster singles and doubles excitation energies (LoFEx-CCSD). *Mol. Phys.*, MQM2016(February):1–10, feb 2017.
- [87] Pablo Baudin, Thomas Kjærgaard, and Kasper Kristensen. CC2 oscillator strengths within the local framework for calculating excitation energies (LoFEx). *J. Chem. Phys.*, 146(14):144107, 2017.
- [88] Pablo Baudin and Kasper Kristensen. Correlated natural transition orbital framework for low-scaling excitation energy calculations (CorNFLEx). *J. Chem. Phys.*, 146:214114, 2017.
- [89] Christof Hättig and Florian Weigend. CC2 excitation energy calculations on large molecules using the resolution of the identity approximation. *J. Chem. Phys.*, 113(13):5154–5161, 2000.
- [90] Christof Hättig and Andreas Köhn. Transition moments and excited-state first-order properties in the coupled-cluster model CC2 using the resolution-of-the-identity approximation. *J. Chem. Phys.*, 117(15):6939–6951, 2002.
- [91] Kasper Kristensen, Patrick Ettenhuber, Janus Juul Eriksen, Frank Jensen, and Poul Jørgensen. The same number of optimized parameters scheme for determining intermolecular interaction energies. *J. Chem. Phys.*, 142(11):114116, 2015.
- [92] Ove Christiansen, Asger Halkier, Henrik Koch, Poul Jørgensen, and Trygve Helgaker. Integral-direct coupled cluster calculations of frequency-dependent polarizabilities, transition probabilities and excited-state properties. *J. Chem. Phys.*, 108(7):2801–2816, 1998.
- [93] Pablo Baudin, José Sánchez Marín, Inmaculada García Cuesta, and Alfredo M J Sánchez de Merás. Calculation of excitation energies from the CC2 linear response theory using Cholesky decomposition. *J. Chem. Phys.*, 140(10):104111, 2014.

- [94] Alok Shukla, Michael Dolg, Hermann Stoll, and Peter Fulde. An ab initio embedded-cluster approach to electronic structure calculations on perfect solids: a Hartree-Fock study of lithium hydride. *Chemical Physics Letters*, 262:213–218, 1996.

Part IV

Index

Index

**GENERAL, 71
**INTEGRAL, 71
**OPENRSP, 70
**RESPONS, 58, 61, 66
.**RESPONS, 83
**WAVE FUNCTIONS, 71
*ALPHA, 61
. *CCRESPONSE, 83, 84
*DENSOPT, 71
*ESGRAD, 67
*Label for property 1, 58
*Label for property 2, 58
*RESPONSE FUNCTION, 70, 78
*SOLVER, 70
..., 71
.BFREQ, 61
.CONVTHR, 71
.EXSTATES, 67
.General response information, 58
.HF, 71
.IMBFREQ, 62
.NEXCIT, 66
.NO SCREEN, 71
.NOFAMILY, 71
.NOGCBASIS, 71
.Specific (optional) information for property 1, 58
.Specific (optional) information for property 2, 58
\$PROPERTY, 70–72, 78
\$RESTART_FLAG, 79
\$WRITE_THRES, 77, 80
MOLECULE, 16–18, 28, 32–34
0.05 0.0, 62
0.1 0.2, 61
1.0e–8, 71
.2ND_ALL, 45
.2ND_LOC, 45
3 6, 67
.3BODY, 42
2, 61, 62, 67
6, 67
.ADMM, 32
.ADMM-FUNC, 32, 33
.ADMM1, 32
.ADMM2, 32
.ADMMP, 33
.ADMMQ, 33
.ADMMS, 32
.ANGINT, 41
.AO_CUTOFF, 93
.AO_SHUKLA, 94
.AOPREC, 59
.AOPRINT, 33
ARH, 45
.ARH DAVID, 45
.ARH FULL, 45
.ARH(LS) DAVID, 45
AUGMENTED ROOTHAAN-HALL, 45
B3LYP, 39

- B3LYP, GAUSSIAN VERSION, 39
- B3LYPG, 39
- B3P86, 39
- B3P86, GAUSSIAN VERSION, 39
- .BAKER, 51, 52
- BASIS SET
 - LIBRARY, 4
- .BASPRINT, 33
- BECKE, 38
- .BFGS, 50, 52
- .BFREQ, 61, 62, 67
- BLYP, 39
- BP86, 39
- BPW91, 39
- BUILDING AN EXECUTABLE, 5

- C, 2
- C PREPROCESSOR, 2
- CAMB3LYP, 40
- .CANONICAL, 81, 85, 86, 88
- .CART-E, 36
- .CARTES, 51, 52
- CARTESIAN COORDINATES, 52
 - .CC2, 81, 86
 - .CCMAXITER, 88
 - .CCS, 85
 - .CCS_NTO_THR, 84
 - .CCSD, 81, 83, 86
 - .CCSD(T), 82, 86
 - .CCSDNOSAFE, 88
 - .CCTHR, 88
 - .CFREQ, 62, 67
 - .CHARGEDIST, 91
 - .CHOLESKY, 45
 - .COARSE, 41
 - .CONSTR, 52
 - .CONFAC, 45
 - .CONTRAC, 45
 - .CONV_STARTGUESS, 94
 - .CONVDYN, 45, 60
 - .CONVTHR, 46, 59
- COORDINATE SYSTEM
 - CARTESIAN COORDINATES, 52
 - REDUNDANT INTERNAL COORDINATES, 53
- CRAY, 2
- .CS00, 44
- .CS00 SHIFT, 44
- .CS00 ZND1, 44
- .CSR, 31

- .D2PAR, 42
- .D3PAR, 42
- .DAMPEDRANGE, 64
- .DAMPEDXCOORD, 64
- .DEBUG, 32
- .DEBUG_MODE, 93
- .DECPRINT, 82
- .DEGENERATE, 64
- .DENS, 90
- .DENSFIT, 33
- .DENSITY, 82
- .DFP, 50, 53
- .DFREQ, 68
- .DFT, 31, 37, 48
- .DFT-D2, 42
- .DFT-D3, 42
- .DFT-D3BJ, 42
- .DFTELS, 41
- .DFTTHR, 41
- .DIIS, 46
- .DISK, 46
- .DISKSOLVER, 46
- .DISPER, 42
- .DTHR, 60

- .EP, 90
- .EXPAND, 46

- .EXPFAC, 46
- .EXSTATES, 66, 68

- .FD_THR, 84
- .FINDRE, 52, 53
- .FINE, 42
- FIRST-ORDER OPTIMIZATION, 52, 53
- .FIXSHIFT, 46
- .FOCKMD, 55
- .FORCE_SPECTRUM, 83
- .FORCEGCBASIS, 31
- FORTRAN 90, 2
- .FOT, 82
- .FREQUENCY, 73
- .FROZENCORE, 88
- .FTUVMAXPRIM, 36

- .GAMMA, 63
- .GAUSSIAN, 65
- .GRADIENT, 82
- .GRID TYPE, 42
- .GRID1, 43
- .GRID2, 43
- .GRID3, 43
- .GRID4, 41, 43
- .GRID5, 43

- .HARDNESS, 43
- .HESVEC, 46
- .HF, 37, 48
- .HFRESTART, 87

- .IMBFREQ, 61, 62, 67
- .IMCFREQ, 62, 67
- .IMDFREQ, 68
- INFO_CROP, 50
- INFO_DIIS, 50
- INFO_LEVELSHIFT, 50
- INFO_LINEQ, 50
- INFO_RH, 50

- INFO_RH_DETAIL, 50
- INFO_STABILITY, 50
- INFO_STABILITY_REDSpace, 50
- Information for property 1, 70
- Information for property 2, 71
- Information for property i, 72
- .INIMOD, 50, 53
- .INIRED, 50, 53
- .INITEV, 50, 53
- .INPUT, 89
- INSTALL DIRECTORY, 4
- INSTALLATION
 - BASIS SET LIBRARY, 4
 - INSTALL DIRECTORY, 4
 - MAKEFILE, 3
 - MAKEFILE.CONFIG, 3
 - SCRATCH SPACE, 4
- INSTALLATION INSTRUCTIONS, 2
- .INTEGRALTHRESHOLD, 88
- .INTERACTIONENERGY, 31

- .KAPPATHR, 82
- .KN-RULE, 74
- KT1, 40
- KT2, 40
- KT3, 40

- LB94, 38
- .LB94, 44
- .LCM, 47
- .LCV, 46
- LDA, 39
- .LEVELSH, 47
- .LINESHAPEPARAM, 65
- .LINSCAPRINT, 33
- LINUX, 2
- .LMAX, 37
- .LOOSE, 51, 53
- .LOOSE MICRO THRESH, 57

- .LOW ACCURACY START, 47
- .LOW RJO00 ACCURACY, 36
- LYP, 38
- MACOSX, 2
- .MACRO IT, 57
- .MACRO_THR, 89
- MAKEFILE, 3
- MAKEFILE.CONFIG, 3, 5
- .MANUAL, 92
- .MASSWE, 55
- .MAX IT, 53
- .MAX ITER, 55
- .MAXELM, 47
- .MAXIT, 47, 59
- .MAXPASSES, 36
- .MAXRED, 59
- .MAXSTEP, 47
- .MCDEXCIT, 64
- .MCDEXSTATES, 64
- .MEMORY, 81, 82, 86
- .MICROVECS, 47
- .MICTHRS, 47
- .MINDAMP, 47
- .MO_SHUKLA, 94
- .MODHES, 51, 53
- .MOLECULE, 92
- .MOLPRINT, 34
- .MP2, 81, 85
- .MPIGROUPSIZE, 83
- .MWVEL, 55
- .NAVAT, 83
- .NEXCIT, 58, 88
- .NF_CUTOFF, 93
- .NO ATERM, 65
- .NO BTERM, 65
- .NO CS, 35
- .NO DAMPED, 65
- .No Level2 Localization, 57
- .NO LONDON, 65
- .NO NONLONDON, 65
- .NO PASS, 36
- .NO PS, 35
- .NO SCREEN, 34
- .NO SIMULATE, 65
- .NO_REDUNDANT, 93
- .NOAV, 47
- .NOFAMILY, 34
- .NOGCBASIS, 31, 48
- .NOJENGINE, 34
- .NOLINK, 34
- .NOMMBU, 37
- .NOONE, 37
- .NOPREC, 48, 59
- .NOPRUN, 43
- .NORMAL, 43
- .NOSEGMENT, 36
- .NOSHIFT, 48
- .NOTABSORBH, 82
- .NSETUV, 36
- .NSTEPS, 65
- .NTO_CCS, 83
- .NTO_THR, 84
- .NVEC, 48
- .NVECFORPEAK, 65
- .OccPipekM(MULL), 57
- .OccPipekMezey, 56
- OLYP, 40
- .Only Loc, 57
- .OPERATOR, 89
- .OPFREQ, 63
- Optional flags, 70
- OPTX, 38
- .ORB, 90
- .Orbital Locality, 57
- .Orbital Plot, 57, 58

- .ORHPOT_STRENGTH, 93
- .OUTPUT, 90
- .OVERLAP-DF-J, 36
- P86, 38
- .PAIR_SOLVER, 59
- .PARI, 34
- PBE, 40
- PBE0, 40
- PBEC, 38
- PBEx, 38
- .PERTURBATION, 73
- .PFM, 56
- .PipekM(MULL), 56
- .PipekMezey, 56, 57
- PORTING, 2
- .PRINT, 53, 54, 72
- .PSB, 50, 54
- .PSFUN, 31, 41
- .PSM, 56
- PW86x, 38
- PW91, 38
- PZ81, 38

- .RADINT, 43
- .REDINT, 51, 53
- REDUNDANT INTERNAL COORDINATES, 53
- Response solver flags, 70
- .RESTART, 48, 51, 86, 87
- .RESEXEC, 60
- .RH, 48
- .RICC2, 86
- .RICCS, 85
- .RIMP2, 81, 85
- .RUNMM, 34

- .SAMESUBSYSTEMS, 31
- .SCALAPACK, 31
- .SCALAPACKBLOCKSIZE, 31
- .SCALAPACKNODES, 31

- SCRATCH SPACE, 4
- .SCREEN, 35, 37
- .SCREENING_THRESHOLD, 93
- .SEGMENT_LENGTH, 32
- .SEPARATE, 92
- SGI, 2
- .SKIPSCFLOOP, 48
- SLATER, 38
- .SNOOP, 86
- .SNOOPMAXDIIS, 87
- .SNOOPMAXITER, 87
- .SNOOPRESTART, 87
- .SNOOPTHR, 86
- .SOEO, 48
- .SOEODIPOLE, 49
- .SOEOGC, 49
- .SOEOMATHR, 49
- .SOEOMITHR, 49
- .SOEOOCC, 48, 49
- .SOEORST, 49
- .SOEOSAVE, 49
- .SOEOSPACE, 48, 49
- .SOEOTRUST, 49
- .SPECTRUM, 89
- .STAB MAXIT, 45
- .STABILITY, 45
- .START, 49
- .SUBSIZE, 88
- .SUBSYSTEMDENSITY, 31
- SVWN, 39
- SVWN3, 39
- .SYM_SOLVER, 59

- .THR_CS, 34, 35
- .THR_PS, 34, 35
- .THRES, 69, 83, 89
- .THRESH, 34
- .TIGHT, 51, 54
- .TIME, 31

.TIMESTEP, 55
.TIMREV, 55
.TLMAX, 37
.TRILEVEL, 52

.ULTRAF, 44
.UNCONT, 36

.VanLenthe, 50
.VELOCI, 55
.VirtPipekM(MULL), 57
.VirtPipekMezey, 56
.VLOOSE, 51, 54
.VTIGHT, 51, 54
VWN3, 38
VWN5, 38

.XOS_OPT_MODEL, 83