SALMON document

Release v.2.0.0

SALMON developers

Jul 21, 2020

CONTENTS

Jul 21, 2020

SALMON is an open-source software based on first-principles time-dependent density functional theory to describe optical responses and electron dynamics in matters induced by light electromagnetic fields.

CHAPTER

ONE

INTRODUCTION

1.1 Overview

SALMON is an open-source computer program for ab-initio quantum-mechanical calculations of electron dynamics at the nanoscale that takes place in various situations of light-matter interactions. It is based on time-dependent density functional theory, solving time-dependent Kohn-Sham equation in real time and real space with norm-conserving pseudopotentials.

SALMON was born by unifying two scientific programs: ARTED, developed by Univ. Tsukuba group, that describes electron dynamics in crystalline solids, and GCEED, developed by Institute for Molecular Science group, that describes electron dynamics in molecules and nanostructures. It can thus describe electron dynamics in both isolated and periodic systems. It can also describe coupled dynamics of electrons and light-wave electromagnetic fields.

To run the program, SALMON requires MPI Fortran/C compiller with LAPACK libraries. SALMON has been tested and optimized to run in a number of platforms, including Linux PC Cluster with x86-64 CPU, supercomputer systems with Fujitsu FX100 and A64FX processors, and supercomputer system with Intel Xeon Phi (Knights Landing).

1.2 SALMON features

In the microscopic scale, SALMON describes electron dynamics in both isolated (molecules and nanostructures) and periodic (crystalline solids) systems, solving time-dependent Kohn-Sham equation in real time and real space with norm-conserving pseudopotential. SALMON first carries out ground-state calculations in the density functional theory to prepare initial configurations. SALMON then calculates electron dynamics induced by applied electric field. Employing a weak impulsive external field, SALMON can be used to calculate linear response properties such as a polarizability of molecules and a dielectric function of crystalline solids. Using pulsed electric fields, SALMON describes electron dynamics in matters induced by intense and ultrashort laser pulses.

SALMON is also capable of describing a propagation of electromagnetic fields of light using finite-difference time-domain method. As a unique feature of SALMON, it is possible to carry out calculations of a coupled dynamics of light electromagnetic fields and electron dynamics simultaneously.

Efficient parallelizations are implemented in the code by dividing spatial grids, orbital index, and k-points. SALMON shows a good scalability when it runs in parallel supercomputers, both for the ground state and the time evolution calculations.

- Ground state calculations
  - Kohn-Sham orbitals and energies
  - density of states
  - projected density of states
  - electron localization function

- Optical properties
  - Oscillator strength distribution (absorption spectrum)
  - dielectric function
- Light-induced electron dynamics
  - time evolution of Kohn-Sham orbitals
  - density, current
  - excitation energy
  - number density of excited carriers
- Propagation of light electromagnetic fields
  - Drude-Lorentz model
  - optical response of metasurfaces
- Simultaneous description of electron dynamics and light pulse propagation
  - light pulse propagation as well as time evolution of Kohn-Sham orbitals
  - energy transfer from pulsed light to electrons

1.3 License

SALMON is available under Apache License version 2.0.

1.4 SALMON at Github

SALMON is developed at GitHub.com

1.5 List of developers

(Alphabetic order)

- Yuta Hirokawa (University of Tsukuba, Japan)
- Kenji Iida (Hokkaido University, Japan)
- Tomohito Otobe (National Institutes for Quantum and Radiological Science and Technology, Japan)
- Shunsuke Sato (University of Tsukuba, Japan)

- Yasushi Shinohara (University of Tokyo, Japan)
- Takashi Takeuchi (University of Tsukuba, Japan)
- Mitsuharu Uemoto (Kobe University, Japan)
- Kazuhiro Yabana (University of Tsukuba, Japan)
- Atsushi Yamada (University of Tsukuba, Japan)
- Shunsuke Yamada (University of Tsukuba, Japan)

1.6 Former developers

- Isabella Floss
- Kazuya Ishimura
- Kyung-Min Lee
- Katsuyuki Nobusada
- Masashi Noda
- Xiao-Min Tong
- Maiku Yamaguchi

1.7 Acknowledgements for SALMON developments

SALMON has been developed by the SALMON developers under supports by Center for Computational Sciences, University of Tsukuba, and National Institute for Quantum and Radiological Science and Technology. SALMON has been supported by Strategic Basic Research Programs, CREST, Japan Science and Technology Agency, under the Grand Number JPMJCR16N5, in the research area of Advanced core technology for creation and practical utilization of innovative properties and functions based upon optics and photonics. SALMON was also supported by Ministry of Education, Culture, Sports and Technology of Japan as a social and scientific priority issue (Creation of new functional devices and high-performance materials to support next-generation industries: CDMSI) to be tackled by using post-K computer.

CHAPTER

TWO

INSTALL AND RUN

2.1 Prerequisites

In this guide, it is assumed that readers have a basic knowledge of Linux and its command line operations. For the installation of SALMON, following packages are required.

- Fortran90/C compiler. SALMON assumes users have one of the following compilers:
  - GCC (Gnu Compiler Collection)
  - Intel Compiler
  - Fujitsu Compiler (at FX100 and A64FX)
- One of the following library packages for linear algebra:
  - Netlib BLAS/LAPACK/ScaLAPACK
  - Intel Math Kernel Library (MKL)
  - Fujitsu Scientific Subroutine Library 2 (SSL-II)
- Build tools:
  - CMake

If you use other compilers, you may need to change build scripts (CMake). See *Additional options in configure.py script*. If no numerical library is installed on your computer system, you may need to install BLAS/LAPACK by yourself. See *Troubleshooting of the Installation Process*.

For the installation of SALMON, we adopt the CMake tools as the first option. If there were any problems to use CMake tools in your environment, you may use the GNU make tools. See *Troubleshooting of the Installation Process*.

2.2 Download

The newest version of SALMON can be downloaded from download page. To extract files from the downloaded file `SALMON-<VERSION>.tar.gz`, type the following command in the command-line:

```
$ tar -zxvf ./salmon-<VERSION>.tar.gz
```

After the extraction, the following directories will be created:

```
SALMON
  |- src          Source codes
  |- example      Samples
```

(continues on next page)

(continued from previous page)

```
  |- cmakefiles   CMake related files
  |- gnumakefiles GNU Makefiles for building
```

2.3 Build and Install

To compile SALMON to create executable the binary files, we adopt to use CMake tools as the first option. In case you fail to build SALMON using CMake in your environment, we may use Gnu Make. See *Build using GNU Makefile*.

2.3.1 Checking CMake availability

First, examine whether CMake is usable in your environment or not. Type the following in Linux command-line:

```
$ cmake --version
```

If CMake is not installed in your system, an error message such as `cmake: command not found` will appear. If CMake is installed on your system, the version number will be shown. To build SALMON, CMake of version 3.14.0 or later is required. If you confirm that CMake of version 3.14.0 or later is installed in your system, proceed to *Build using CMake*. However, we realize that old versions of CMake are installed in many systems. If CMake is not installed or CMake of older versions is installed in your system, you need to install the new version by yourself. It is a simple procedure and explained below.

2.3.2 Installation of CMake (pre-compiled binary of Linux)

CMake is a cross-platform build tool. The simplest way to make CMake usable in your environment is to get the binary distribution of CMake from the download page. (The file name of the binary distribution will be `cmake-<VERSION>-<PLATFORM>.tar.gz`). In standard Linux environment, a file for the platform of Linux x86_64 will be appropriate.

To download the file, proceed as follows: We assume that you are in the directory that you extracted files from the downloaded file of SALMON, and that you will use the version 3.16.8. First get the URL of the download link from your browser, and use `wget` command in your Linux command-line:

```
$ wget https://cmake.org/files/v3.16/cmake-3.16.8-Linux-x86_64.tar.gz
```

Next, unpack the archive by:

```
$ tar -zxvf cmake-3.16.8-Linux-x86_64.tar.gz
```

and you will have the binary `make-3.16.8-Linux-x86_64/bin/cmake` in your directory.

To make the `cmake` command usable in your command-line, you need to modify the environment variable `$PATH` so that the executable of CMake are settled inside the directory specified in your `$PATH`. If you use the bash shell, you need to modify the file `~/.bashrc` that specifies the `$PATH` variable. It can be done by typing the following command in your login directory:

```
$ export PATH=<SALMON_INSTALLATION_DIRECTORY>/cmake-3.16.8-Linux-x86_64/bin:$PATH
```

and then reload the configuration by typing:

```
$ source ~/.bashrc
```

See *Installation of CMake* describes Other way of the installation.

2.3.3 Build using CMake

Confirming that CMake of version 3.14.0 or later can be usable in your environment, proceed the following steps. We assume that you are in the directory SALMON.

1. Create a new temporary directory `build` and move to the directory:

```
$ mkdir build
$ cd build
```

2. Execute the python script "configure.py" and then make:

```
$ python ../configure.py --arch=ARCHITECTURE --prefix=../
$ make
$ make install
```

In executing the python script, you need to specify `ARCHITECTURE` that indicates the architecture of the CPU in your computer system such as `intel-avx`. The options of the `ARCHITECUTRE` are as follows:

arch	Detail	Compiler	Numerical Library
intel-knl	Intel Knights Landing	Intel Compiler	Intel MKL
intel-knc	Intel Knights Corner	Intel Compiler	Intel MKL
intel-avx	Intel Processer (Ivy-, Sandy-Bridge)	Intel Compiler	Intel MKL
intel-avx2	Intel Processer (Haswell, Broadwell ..)	Intel Compiler	Intel MKL
intel-avx512	Intel Processer (Skylake-SP)	Intel Compiler	Intel MKL
fujitsu-fx100	FX100 Supercomputer	Fujitsu Compiler	SSL-II
fujitsu-a64fx-ea	A64FX processor (Fugaku, FX1000, FX700)	Fujitsu Compiler	SSL-II

If the build is successful, you will get a file `salmon` at the top-level build directory.

2.4 Files necessary to run SALMON

To run SALMON, at least two kinds of files are required for any calculations. One is an input file with the filename extension `*.inp` that should be read from the standard input `stdin`. This file should be prepared in the Fortran90 namelist format. Pseudopotential files of relevant elements are also required. Depending on your purpose, some other files may also be necessary. For example, coordinates of atomic positions of the target material may be either written in the input file or prepared as a separate file.

2.4.1 Pseudopotentials

SALMON utilizes norm-conserving pseudpotentials. You may find pseudopotentials of some elements in the samples prepared in *Exercises*. In SALMON, several formats of pseudopotentials may be usable. Pseudopotentials with an extension `.fhi` can be obtained from the website listed below. (This is a part of previous atomic data files for the ABINIT code.)

Pseudopotential	Website
Pseudopotentials for the ABINIT code	https://www.abinit.org/sites/default/files/PrevAtomicData/psp-links/psp-links/lda_fhi

Filenames of the pseudopotentials should be written in the input file.

2.4.2 input file

Input files are composed of several blocks of namelists:

```
&namelist1
  variable1 = int_value
  variable2 = 'char_value'
/
&namelist2
  variable1 = real8_value
  variable2 = int_value1, int_value2, int_value3
/
```

A block of namelists starts with `&namelist` line and ends with `/` line. The blocks may appear in any order.

Between two lines of `&namelist` and `/`, descriptions of variables and their values appear. Note that many variables have their default values so that it is not necessary to give values for all variables. Descriptions of the variables may appear at any position if they are between `&namelist` and `/`.

SALMON describes electron dynamics in systems with both isolated and periodic boundary conditions. The boundary condition is specified by the variable `iperiodic` in the namelist `&system`.

Calculations are usually achieved in two steps; first, the ground state calculation is carried out and then electron dynamics calculations in real time is carried out. A choice of the calculation mode or theory in the calculation is specified by the variable `theory` in the namelist `&calculation`. In the typical way, the ground state calculation based on DFT is first carried out specifying `theory = 'dft'`. Then the real-time electron dynamics calculation based on TDDFT is carried out specifying `theory = 'tddft_pulse'`.

In *Exercises*, we prepare six exercises that cover typical calculations feasible by SALMON. We prepare explanations of the input files of the exercises that will help to prepare input files of your own interests.

There are more than 20 groups of namelists. A complete list of namelist variables is given in the file `SALMON/manual/input_variables.md`. Namelist variables that are used in our exercises are explained at *Input Keywords for Exercises*.

2.5 Run SALMON

Before running SALMON, the following preparations are required as described above: The executable file of `salmon` should be built from the source file of SALMON. An input file `inputfile.inp` and pseudopotential files should also be prepared.

The execution of the calculation can be done as follows: In single process environment, type the following command:

```
$ salmon < inputfile.inp > fileout.out
```

In multiprocess environment in which the command to execute parallel calculations using MPI is `mpiexec`, type the following command:

```
$ mpiexec -n NPROC salmon < inputfile.inp > fileout.out
```

where NPROC is the number of MPI processes that you will use.

The execution command and the job submission procedure depends much on local environment. We summarize general conditions to execute SALMON:

- SALMON runs in both single-process and multi-process environments using MPI.
- executable files are prepared as `salmon` in the standard build procedure.

- to start calculations, `inputfile.inp` should be read through `stdin`.

2.6 Tips for large-scale calculation

We explain below some tips that will be useful to improve performance when you carry out large scale simulations using world top-level supercomputers. Therefore, the following contents will only be useful only for limited users.

2.6.1 MPI process distribution

SALMON provides three variables to determine the process distribution/allocation.

- `nproc_k`
- `nproc_ob`
- `nproc_rgrid(3)`

In SALMON, the process distribution is determined automatically as default. However, in many situations, an explicit assignment of the process distribution will provide a better performance than the default setting.

We recommend to distribute the processes as follows,

If you use k-points (the number of k-points is greater than 1) and the number of the real-space grid (`num_rgrid`) is not very large (about 16^3):

- First, assign many processes to `nproc_k`.
- Then, assign the remaining processes to `nproc_ob`.
- Not dividing the spatial grid, `nproc_rgrid = 1, 1, 1`.

Else:

- First, assign the processes to `nproc_ob`.
- Then, assign the remaining processes to `nproc_rgrid`.
  - If real-space grid size (`num_rgrid(1:3) = al(1:3) / dl(1:3)`) is equal to or larger than about 64^3,

  you should find a balanced distribution between `nproc_rgrid` and `nproc_ob`.

2.6.2 Improve the performance of the eigenvalues solver

In DFT calculations of large systems, subspace diagonalization becomes the performance bottleneck in the entire calculation. Therefore, it is important to use a parallel eigenvalues solver. In SALMON, a LAPACK routine without parallelization is used for the diagonalization as default. As parallelized solvers, ScaLAPACK and EigenExa are usable. To use them, it is necessary to rebuild SALMON enabling ScaLAPACK/EigenExa. You can find the instruction in *Install and Run*.

To execute SALMON using ScaLAPACK/EigenExa, either `yn_scalapack = 'y'` or `yn_eigenexa = 'y'` should be included in the inputfile:

```
&parallel
  yn_scalapack = 'y'          ! use ScaLAPACK for diagonalization
  !yn_eigenexa  = 'y'         ! use EigenExa
  yn_scalapack_red_mem = 'y' ! to reduce the memory consumption
/
```

ScaLAPACK/EigenExa solves the eigenvalue problem with `nproc_ob` process distribution. If `nproc_ob = 1`, ScaLAPACK/EigenExa will perform in the same way as the LAPACK library.

2.6.3 Improve the performance of Hartree solver

For periodic systems, a Fourier transformation is used to solve the Poisson equation (to calculate the Hartree potential). In SALMON, a simple Fourier transformation without Fast Fourier Transformation (FFT) is used as default. In SALMON, a parallelized FFT foutine, FFTE, is usable and works efficiently for large systems. In using FFTE, the following conditions should be satisfied:

```
num_rgrid(1) mod nproc_rgrid(2) = 0
num_rgrid(2) mod nproc_rgrid(2) = 0
num_rgrid(2) mod nproc_rgrid(3) = 0
num_rgrid(3) mod nproc_rgrid(3) = 0

In addition, the prime factors for the number of real-space grid of each direction␣
↪(num_rgrid(1:3)) must be a combination of 2, 3 or 5.
```

To use FFTE, `yn_ffte = 'y'` should be included in the input file:

```
&parallel
  yn_ffte = 'y'
/
```

2.6.4 Improve IO performance (write/read wavefunction)

Almost all supercomputer systems provide distributed filesystems such as Lustre. Distributed filesystems are equipped with a meta-data server (MDS) and an object-storage server (OST). The OST stores real user data files, and the MDS stores the address of the user date files in the OST. When accessing to the data files in the OST, the process send a query about the OST address to MDS. Then, a network contention may occur in the query process.

In most implementations of the filesystem, the MDS that replies to the query is determined by the directory structure. For a calculation in which k-point is not used, `method_wf_distributor` and `nblock_wf_distribute` are prepared to reduce the network contention:

```
&control
  method_wf_distributor = 'slice' ! every orbital function is stored as a single file.
  nblock_wf_distribute  = 32      ! files of 32 orbital functions are stored in one␣
↪directory.
/
```

2.6.5 Improve the communication performance for mesh-torus network system

Large-scale supercomputers often adopt a mesh-torus network system such as Cray dragon-fly and Fujitsu Tofu to achieve high scalability with relatively low cost. In SALMON, a special MPI process distribution (communicator creation rule) is prepared to improve the performance in large-scale mesh-torus network systems.

Currently, we provide the communicator creation rule for "Supercomputer Fugaku", which is developed by RIKEN R-CCS and Fujitsu limited. Fugaku is equipped with a 6-D mesh-torus network which is called "Tofu-D". Users may control it as a 3-D logical network. SALMON utilizes 5-D array (wavefunction(x, y, z, orbital, k-point)) as a domain for parallelization. We create a map that connects the 3-D network to the 5-D array distribution.

We introduce the following variables and conditons to assign the 3-D mesh-torus network to the 5-D array distribution:

```
PW           = nproc_ob * nproc_k
(PX, PY, PZ) = nproc_rgrid
PPN          = '# of process per node' (we recommend the value 4 in Fugaku)

Requested process shape: (PX, PY, PZ, PW)
Tofu-D network    shape: (TX, TY, TZ)
Actual process    shape: (TX * PPN, TY, TZ)

if (process_allocation == 'grid_sequential'):
  PW  = PW1 * PW2 * PW3
  PW1 = (TX * PPN) / PX
  PW2 = TY         / PY
  PW3 = TZ         / PZ
  TX  = (PX * PW1) / PPN
  TY  = PY * PW2
  TZ  = PZ * PW3

else if (process_allocation == 'orbital_sequential'):
  PX  = PX1 * PX2 * PX3
  PX1 = (TX * PPN) / PW
  PX2 = TY         / PY
  PX3 = TZ         / PZ
  TX  = (PW * PX1) / PPN
  TY  = PY * PX2
  TZ  = PZ * PX3
```

From these conditions, you can determine the suitable process distribution and the Tofu-D network shape (compute node shape). `process_allocation` input variable controls the order of the process distribution. It indicates which communications should be executed in closer processes.

- `process_allocation = 'grid_sequential'`
  - `(PX, PY, PZ, PW)`, `nproc_rgrid` major ordering
  - improves `nproc_rgrid` related communication performance
  - communicator: `s_parallel_info::icomm_r, icomm_x, icomm_y, icomm_z, icomm_xy`
  - suitable `theory`: `'dft'` and `'dft_md'`
- `process_allocation = 'orbital_sequential'`
  - `(PW, PY, PZ, PX)`, `nproc_ob` major ordering
  - improves `nproc_ob` related communication performance
  - communicator: `s_parallel_info::icomm_o and icomm_ko`
  - suitable `theory`: `'tddft_response', 'tddft_pulse', 'single_scale_maxwell_tddft'` and `'multi_scale_maxwell_tddft'`

2.7 Appendix

2.7.1 Additional options in configure.py script

Manual specifications of compiler and environment variables

In executing `configure.py`, you may manually specify compiler and environment variables instead of specifying the architecture, for example:

```
$ python ../configure.py FC=mpiifort CC=mpiicc FFLAGS="-xAVX" CFLAGS="-restrict -xAVX"
```

The major options of `configure.py` are as follows:

Commandline switch	Detail
-a ARCH, –arch=ARCH	Target architecture
–enable-mpi, –disable-mpi	enable/disable MPI parallelization
–enable-scalapack, –disable-scalapack	enable/disable computations with ScaLAPACK library
–enable-eigenexa, –disable-eigenexa	enable/disable computations with RIKEN R-CCS EigenExa library
–enable-libxc, –disable-libxc	enable/disable computations with Libxc library
–with-lapack	specified LAPACK/ScaLAPACK installed directory
–with-libxc	specified Libxc installed directory
–debug	enable debug build
–release	enable release build
FC, FFLAGS	User-defined Fortran Compiler, and the compiler options
CC, CFLAGS	User-defined C Compiler, and the compiler options

In the build procedure by CMake, they search the following libraries. If the libraries don't found in the path that is specified by environment variables, they will build the required libraries automatically.

- Netlib LAPACK (includes BLAS), and ScaLAPACK
  - We will download and build the Netlib libraries as the typical implementation.
  - http://www.netlib.org/lapack/
  - http://www.netlib.org/scalapack/
- Libxc
  - https://www.tddft.org/programs/libxc/

EigenExa will download and build automatically even if the library is installed to your machine.

Build for single process calculations

When using the `--arch` option, MPI parallelization is enabled as default in almost case. If you use a single processor machine, explicitly specify `--disable-mpi` in executing the python script:

```
$ python ../configure.py --arch=<ARCHITECTURE> --disable-mpi
```

Build by user-specified compiler

If you want that specify the compiler, set the `FC` and `CC` flags in executing the python script:

```
$ python ../configure.py FC=gfortran CC=gcc
```

When not using the `--arch` option, MPI parallelization is disabled as default.

2.7.2 Build using GNU Makefile

If CMake build fails in your environment, we recommend you to try to use Gnu Make for the build process. First, enter the directory `gnumakefiles`:

```
$ cd SALMON/gnumakefiles
```

In the directory, `Makefile` files are prepared for several architectures:

- gnu-mpi
- intel-mpi
- gnu-without-mpi
- intel-without-mpi

`Makefile` files with `*-without-mpi` indicate that they are for single processor environment. Choose `Makefile` appropriate for your environment, and execute the make command:

```
$ make -f Makefile.PLATFORM
```

If the make proceeds successful, a binary file is created in the directory `SALMON/bin/`.

2.8 Troubleshooting of the Installation Process

2.8.1 Installation of CMake

The CMake is a cross-platform build tool. In order to build the SALMON from the source code, the CMake of version 3.14.0 or later is required. You may install it following one of the three instructions below.

Installation by package manager

If your system has a built-in package manager, you may conveniently install the CMake tools as below:

Debian/Ubuntu Linux

```
sudo apt-get install cmake
```

Fedora Linux/CentOS

```
sudo yum install cmake
```

openSUSE Linux

```
sudo zypper install cmake
```

Installation from source code

You can get the source code distribution from the download page. In this time, we will use the cmake version 3.16.8 as an example. Download the archive by `wget` comamnd and unpack it as below:

```
wget https://cmake.org/files/v3.16/cmake-3.16.8.tar.gz
tar -zxvf cmake-3.16.8.tar.gz
```

And, move to the unpacked directory and build.

```
cd cmake-3.16.8
./configure --prefix=INSTALLATION_DIRECTORY
make
make install
```

(replace `INSTALLATION_DIRECTORY` to your installation directory.)

Next, to utilize the `cmake` command, it is required that the executable are settled inside the directory specified in your `$PATH`. If you use the bash shell, edit `~/.bashrc` and append the line:

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

and reload the configuration:

```
source ~/.bashrc
```

CHAPTER

THREE

EXERCISES

3.1 Getting started

Welcome to SALMON Exercises!

In these exercises, we explain the use of SALMON from the very beginning, taking a few samples that cover applications of SALMON in several directions. We assume that you are in the computational environment of UNIX/Linux OS. First you need to download and install SALMON in your computational environment. If you have not yet done it, do it following the instruction, download and *Install and Run*.

As described in *Install and Run*, you are required to prepare at least an input file and pseudopotential files to run SALMON. In the following, we present input files for several sample calculations and provide a brief explanation of the input keywords that appear in the input files. You may modify the input files to execute for your own calculations. Pseudopotential files of elements that appear in the samples are also attached. We also present explanations of main output files.

We present 10 exercises.

First 3 exercises (Exercise-1 ~ 3) are for an isolated molecule, acetylene C2H2. If you are interested in learning electron dynamics calculations in isolated systems, please look into these exercises. In SALMON, we usually calculate the ground state solution first. This is illustrated in *Exercise-1*. After finishing the ground state calculation, two exercises of electron dynamics calculations are prepared. *Exercise-2* illustrates the calculation of linear optical responses in real time, obtaining polarizability and photoabsorption of the molecule. *Exercise-3* illustrates the calculation of electron dynamics in the molecule under a pulsed electric field.

Next 3 exercises (Exercise-4 ~ 6) are for a crystalline solid, silicon. If you are interested in learning electron dynamics calculations in extended periodic systems, please look into these exercises. *Exercise-4* illustrates the ground state solution of the crystalline silicon. *Exercise-5* illustrates the calculation of linear response properties of the crystalline silicon to obtain the dielectric function. *Exercise-6* illustrates the calculation of electron dynamics in the crystalline silicon induced by a pulsed electric field.

Exercise-7 is for an irradiation and a propagation of a pulsed light in a bulk silicon, coupling Maxwell equations for the electromagnetic fields of the pulsed light and the electron dynamics in the unit cells. This calculation is quite time-consuming and is recommended to execute using massively parallel supercomputers. *Exercise-7* illustrates the calculation of a pulsed, linearly polarized light irradiating normally on a surface of a bulk silicon.

Next 2 exercises (Exercise-8 ~ 9) are for geometry optimization and Ehrenfest molecular dynamics based on the TDDFT method for an isolated molecule, acetylene C2H2. *Exercise-8* illustrates the geometry optimization in the ground state. *Exercise-9* illustrates the Ehrenfest molecular dynamics under the pulsed electric field.

Exercise-10 are for an metallic nanosphere described by dielectric function. The calculation method is the Finite-Difference Time-Domain (FDTD). *Exercise-10* illustrates the electromagnetic analysis of the metallic nanosphere under a pulsed electric field.

3.2 C2H2 (isolated molecules)

3.2.1 Exercise-1: Ground state of C2H2 molecule

In this exercise, we learn the calculation of the ground state of acetylene (C2H2) molecule, solving the static Kohn-Sham equation. This exercise will be useful to learn how to set up calculations in SALMON for any isolated systems such as molecules and nanoparticles.

Input files

To run the code, following files in samples are used:

file name	description
C2H2_gs.inp	input file that contains input keywords and their values
C_rps.dat	pseodupotential file for carbon atom
H_rps.dat	pseudopotential file for hydrogen atom

In the input file *C2H2_gs.inp*, input keywords are specified. Most of them are mandatory to execute the ground state calculation. This will help you to prepare an input file for other systems that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 01: Ground state of C2H2 molecule                                    ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html     ␣
↪    !
! * Input format consists of group of keywords like:                            ␣
↪    !
!     &group                                                                    ␣
↪    !
!       input keyword = xxx                                                     ␣
↪    !
!     /                                                                         ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                   ␣
↪    !
!################################################################################
↪###!

&calculation
  !type of theory
  theory = 'dft'
/

&control
  !common name of output files
  sysname = 'C2H2'
/
```

(continues on next page)

(continued from previous page)

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/

&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution ---------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !-------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/

&scf
  !maximum number of scf iteration and threshold of convergence
  nscf      = 300
  threshold = 1.0d-9
/

&analysis
  !output of all orbitals, density,
  !density of states, projected density of states,
  !and electron localization function
  yn_out_psi  = 'y'
```

(continues on next page)

(continued from previous page)

```
  yn_out_dns  = 'y'
  yn_out_dos  = 'y'
  yn_out_pdos = 'y'
  yn_out_elf  = 'y'
/

&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ------------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------------!
/
```

We present their explanations below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'dft'
/
```

This indicates that the ground state calculation by DFT is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

`yn_periodic = 'n'` indicates that the isolated boundary condition will be used in the calculation. `al(1:3) = 16.0d0, 16.0d0, 16.0d0` specifies the lengths of three sides of the rectangular parallelepiped where the grid points are prepared. `nelem = 2` and `natom = 4` indicate the number of elements and the number of atoms in the system, respectively. `nelec = 10` indicate the number of valence electrons in the system. `nstate = 6` indicates the number of Kohn-Sham orbitals to be solved. Since the present code assumes that the system is spin saturated, `nstate` should be equal to or larger than `nelec/2`. See *&system in Inputs* for more information.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/
```

Parameters related to atomic species and pseudopotentials. `file_pseudo(1) = './C_rps.dat'` indicates the filename of the pseudopotential of element. `izatom(1) = 6` specifies the atomic number of the element. `lloc_ps(1) = 1` specifies the angular momentum of the pseudopotential that will be treated as local.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/
```

`dl(1:3) = 0.25d0, 0.25d0, 0.25d0` specifies the grid spacings in three Cartesian directions. See *&rgrid in Inputs* for more information.

&scf

Mandatory: nscf, threshold

```
&scf
  !maximum number of scf iteration and threshold of convergence
  nscf      = 300
  threshold = 1.0d-9
/
```

`nscf` is the number of scf iterations. The scf loop in the ground state calculation ends before the number of the scf iterations reaches `nscf`, if a convergence criterion is satisfied. `threshold = 1.0d-9` indicates threshold of the convergence for scf iterations.

&analysis

Mandatory: none

If the following input keywords are added, the output files are created after the calculation.

```
&analysis
  yn_out_psi  = 'y'
  yn_out_dns  = 'y'
  yn_out_dos  = 'y'
  yn_out_pdos = 'y'
  yn_out_elf  = 'y'
/
```

&atomic_coor

Mandatory: atomic_coor or atomic_red_coor (it may be provided as a separate file)

```
&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ----------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-----------------------------------------------------------------!
/
```

Cartesian coordinates of atoms. The first column indicates the element. Next three columns specify Cartesian coordinates of the atoms. The number in the last column labels the element.

Output files

After the calculation, following output files and a directory are created in the directory that you run the code,

name	description
C2H2_info.data	information on ground state solution
C2H2_eigen.data	1 particle energies
C2H2_k.data	k-point distribution (for isolated systems, only gamma point is described)
data_for_restart	directory where files used in the real-time calculation are contained
psi_ob1.cube, *psi_ob2.cube*, ...	electron orbitals
dns.cube	a cube file for electron density
dos.data	density of states
pdos1.data, *pdos2.data*, ...	projected density of states
elf.cube	electron localization function (ELF)
PS_C_KY_n.dat	information on pseodupotential file for carbon atom
PS_H_KY_n.dat	information on pseodupotential file for hydrogen atom

You may download the above files (zipped file, except for the directory *data_for_restart*) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/01_C2H2_gs.zip
(zipped output files)

Main results of the calculation such as orbital energies are included in *C2H2_info.data*. Explanations of the *C2H2_info.data* and other output files are below:

C2H2_info.data

Calculated orbital and total energies as well as parameters specified in the input file are shown in this file.

C2H2_eigen.data

1 particle energies.

```
#esp: single-particle energies (eigen energies)
#occ: occupation numbers, io: orbital index
# 1:io, 2:esp[eV], 3:occ
```

C2H2_k.data

k-point distribution(for isolated systems, only gamma point is described).

```
# ik: k-point index
# kx,ky,kz: Reduced coordinate of k-points
# wk: Weight of k-point
# 1:ik[none] 2:kx[none] 3:ky[none] 4:kz[none] 5:wk[none]
# coefficients (2*pi/a [a.u.]) in kx, ky, kz
```

psi_ob1.cube, psi_ob2.cube, ...

Cube files for electron orbitals. The number in the filename indicates the index of the orbital atomic unit is adopted in all cube files.

dns.cube

A cube file for electron density.

dos.data

A file for density of states. The units used in this file are affected by the input parameter, `unit_system` in `&unit`.

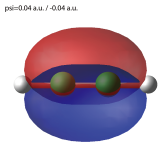
elf.cube

A cube file for electron localization function (ELF).

We show several image that are created from the output files.

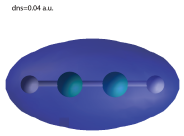
- **Highest occupied molecular orbital (HOMO)**

  The output files *psi_ob1.cube*, *psi_ob2.cube*, ... are used to create the image.



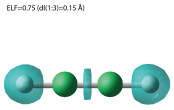
- **Electron density**

  The output files *dns.cube*, ... are used to create the image.



- **Electron localization function**

  The output files *elf.cube*, ... are used to create the image.



3.2.2 Exercise-2: Polarizability and photoabsorption of C2H2 molecule

In this exercise, we learn the linear response calculation in the acetylene (C2H2) molecule, solving the time-dependent Kohn-Sham equation. The linear response calculation provides the polarizability and the oscillator strength distribution of the molecule. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-1*. In the calculation, an impulsive perturbation is applied to all electrons in the C2H2 molecule along the molecular axis which we take z axis. Then a time evolution calculation is carried out without any external fields. During the calculation, the electric dipole moment is monitored. After the time evolution calculation, a time-frequency Fourier transformation is carried out for the electric dipole moment to obtain the frequency-dependent polarizability. The imaginary part of the frequency-dependent polarizability is proportional to the oscillator strength distribution and the photoabsorption cross section.

Input files

To run the code, the input file *C2H2_rt_response.inp* that contains input keywords and their values for the linear response calculation is required. The directory *restart* that is created in the ground state calculation as *data_for_restart* and pseudopotential files are also required. The pseudopotential files should be the same as those used in the ground state calculation. The input files are in samples.

name	description
C2H2_rt_response.inp	input file that contains input keywords and their values
C_rps.dat	pseodupotential file for carbon
H_rps.dat	pseudopotential file for hydrogen
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

In the input file *C2H2_rt_response.inp*, input keywords are specified. Most of them are mandatory to execute the linear response calculation. This will help you to prepare the input file for other systems that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!###############################################################################
↪###!
! Excercise 02: Polarizability and photoabsorption of C2H2 molecule                ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html       ␣
↪    !
! * Input format consists of group of keywords like:                              ␣
↪    !
!     &group                                                                      ␣
↪    !
!       input keyword = xxx                                                       ␣
↪    !
!     /                                                                           ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                     ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * Copy the ground state data directory('data_for_restart') (or make symbolic link) ␣
↪    !
!   calculated in 'samples/exercise_01_C2H2_gs/' and rename the directory to 'restart/
↪'  !
!   in the current directory.                                                     ␣
↪    !
!###############################################################################
↪###!

&calculation
  !type of theory
  theory = 'tddft_response'
/

&control
  !common name of output files
  sysname = 'C2H2'
/

&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

(continues on next page)

(continued from previous page)

```
&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 1.25d-3
  nt = 5000
/

&emfield
  !envelope shape of the incident pulse('impulse': impulsive field)
  ae_shape1 = 'impulse'

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
```

(continues on next page)

(continued from previous page)

```
  !-------------------------------------------------------------------!
/

&analysis
  !energy grid size and number of energy grids for output files
  de      = 1.0d-2
  nenergy = 3000
/

&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

We present their explanations below:

Required and recommended variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'tddft_response'
/
```

This indicates that the real time (RT) calculation to obtain response function is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input file. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: iperiodic, al, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&system in Exercise-1*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&pseudo in Exercise-1*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !spatial grid spacing(x,y,z)
```

(continues on next page)

(continued from previous page)

```
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/
```

`dl(1:3) = 0.25d0, 0.25d0, 0.25d0` specifies the grid spacings in three Cartesian directions. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 1.25d-3
  nt = 5000
/
```

`dt=1.25d-3` specifies the time step of the time evolution calculation. `nt=5000` specifies the number of time steps in the calculation.

&emfield

Mandatory: ae_shape1

```
&emfield
  !envelope shape of the incident pulse('impulse': impulsive field)
  ae_shape1 = 'impulse'

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

`ae_shape1 = 'impulse'` indicates that a weak impulse is applied to all electrons at *t=0*. `epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0` specify a unit vector that indicates the direction of the impulse. See *&emfield in Inputs* for details.

&atomic_coor

Mandatory: atomic_coor or atomic_red_coor (it may be provided as a separate file)

```
&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-----------------------------------------------------------!
/
```

Cartesian coordinates of atoms. The first column indicates the element. Next three columns specify Cartesian coordinates of the atoms. The number in the last column labels the element. They must be the same as those in the ground state calculation.

Output files

After the calculation, following output files are created in the directory that you run the code,

file name	description
C2H2_response.data	polarizability and oscillator strength distribution as functions of energy
C2H2_rt.data	components of change of dipole moment (electrons/plus definition) and total dipole moment (electrons/minus + ions/plus) as functions of time
C2H2_rt_energy.data	components of total energy and difference of total energy as functions of time
PS_C_KY_n.dat	information on pseodupotential file for carbon atom
PS_H_KY_n.dat	information on pseodupotential file for hydrogen atom

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/02_C2H2_lr.zip
(zipped output files)

Explanations of the output files are below:

C2H2_response.data

Time-frequency Fourier transformation of the dipole moment gives the polarizability of the system. Then the strength function is calculated.

```
# Fourier-transform spectra:
# alpha: Polarizability
# df/dE: Strength function
# 1:Energy[eV] 2:Re(alpha_x)[Augstrom^2/V] 3:Re(alpha_y)[Augstrom^2/V] 4:Re(alpha_
↪z)[Augstrom^2/V] 5:Im(alpha_x)[Augstrom^2/V] 6:Im(alpha_y)[Augstrom^2/V] 7:Im(alpha_
↪z)[Augstrom^2/V] 8:df_x/dE[none] 9:df_y/dE[none] 10:df_z/dE[none]
```

C2H2_rt.data

Results of time evolution calculation for vector potential, electric field, and dipole moment.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# ddm_e: Change of dipole moment (electrons/plus definition)
# dm: Total dipole moment (electrons/minus + ions/plus)
# 1:Time[fs] 2:Ac_ext_x[fs*V/Angstrom] 3:Ac_ext_y[fs*V/Angstrom] 4:Ac_ext_z[fs*V/
↪Angstrom] 5:E_ext_x[V/Angstrom] 6:E_ext_y[V/Angstrom] 7:E_ext_z[V/Angstrom] 8:Ac_
↪tot_x[fs*V/Angstrom] 9:Ac_tot_y[fs*V/Angstrom] 10:Ac_tot_z[fs*V/Angstrom] 11:E_tot_
↪x[V/Angstrom] 12:E_tot_y[V/Angstrom] 13:E_tot_z[V/Angstrom] 14:ddm_e_x[Angstrom]␣
↪15:ddm_e_y[Angstrom] 16:ddm_e_z[Angstrom] 17:dm_x[Angstrom] 18:dm_y[Angstrom] 19:dm_
↪z[Angstrom]
```

C2H2_rt_energy.data

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
```

(continues on next page)

(continued from previous page)

```
# Eall0: Initial energy
# 1:Time[fs] 2:Eall[eV] 3:Eall-Eall0[eV]
```

3.2.3 Exercise-3: Electron dynamics in C2H2 molecule under a pulsed electric field

In this exercise, we learn the calculation of the electron dynamics in the acetylene (C2H2) molecule under a pulsed electric field, solving the time-dependent Kohn-Sham equation. As outputs of the calculation, such quantities as the total energy and the electric dipole moment of the system as functions of time are calculated. This tutorial should be carried out after finishing the ground state calculation that was explained in *Exercise-1*. In the calculation, a pulsed electric field that has cos^2 envelope shape is applied. The parameters that characterize the pulsed field such as magnitude, frequency, polarization direction, and carrier envelope phase are specified in the input file.

Input files

To run the code, following files in samples are used. The directory *restart* is created in the ground state calculation as *data_for_restart*. Pseudopotential files are already used in the ground state calculation. Therefore, *C2H2_rt_pulse.inp* that specifies input keywords and their values for the pulsed electric field calculation is the only file that the users need to prepare.

file name	description
C2H2_rt_pulse.inp	input file that contain input keywords and their values.
C_rps.dat	pseodupotential file for carbon
H_rps.dat	pseudopotential file for hydrogen
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

In the input file *C2H2_rt_pulse.inp*, input keywords are specified. Most of them are mandatory to execute the calculation of electron dynamics induced by a pulsed electric field. This will help you to prepare the input file for other systems and other pulsed electric fields that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!================================================================================
↪===!
! Excercise 03:  Electron dynamics in C2H2 molecule under a pulsed electric field
↪   !
!--------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').
↪   !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html
↪   !
! * Input format consists of group of keywords like:
↪   !
!     &group
↪   !
!       input keyword = xxx
↪   !
!     /
↪   !
!   (see chapter: 'List of all input keywords' in the manual)
↪   !
```

(continues on next page)

(continued from previous page)

```
!-------------------------------------------------------------------------------
↪---!
! * Copy the ground state data directory('data_for_restart') (or make symbolic link) ␣
↪   !
!   calculated in 'samples/exercise_01_C2H2_gs/' and rename the directory to 'restart/
↪'  !
!   in the current directory.                                                    ␣
↪   !
!###############################################################################
↪###!

&calculation
  !type of theory
  theory = 'tddft_pulse'
/

&control
  !common name of output files
  sysname = 'C2H2'
/

&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/

&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !--------------------------------------------------!
/
```

(continues on next page)

(continued from previous page)

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 1.25d-3
  nt = 5000
/

&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 6.00d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 9.28d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/

&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !--------------------------------------------------------------!
/
```

We present explanations of the input keywords that appear in the input file below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'tddft_pulse'
/
```

This indicates that the real time (RT) calculation for a pulse response is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input file. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, nelem, natom, nelectron, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 16.0d0, 16.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&system in Exercise-1*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&pseudo in Exercise-1*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/
```

`dl(1:3) = 0.25d0, 0.25d0, 0.25d0` specifies the grid spacings in three Cartesian directions. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 1.25d-3
  nt = 5000
/
```

`dt = 1.25d-3` specifies the time step of the time evolution calculation. `nt = 5000` specifies the number of time steps in the calculation.

&emfield

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 6.00d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 9.28d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

These input keywords specify the pulsed electric field applied to the system.

`ae_shape1 = 'Ecos2'` indicates that the envelope of the pulsed electric field has a *cos*^2 shape.

`I_wcm2_1 = 1.00d8` specifies the maximum intensity of the applied electric field in unit of W/cm^2.

`tw1 = 6.00d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 9.28d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0` specifies the real part of the unit polarization vector of the pulsed electric field. Using the real polarization vector, it describes a linearly polarized pulse.

`phi_cep1 = 0.75d0` specifies the carrier envelope phase of the pulse. As noted above, 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish.

See *&emfield in Inputs* for details.

&atomic_coor

Mandatory: atomic_coor or atomic_red_coor (it may be provided as a separate file)

```
&atomic_coor
  !cartesian atomic coodinates
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !--------------------------------------------------------------!
/
```

Cartesian coordinates of atoms. The first column indicates the element. Next three columns specify Cartesian coordinates of the atoms. The number in the last column labels the element. They must be the same as those in the ground state calculation.

Output files

After the calculation, following output files are created in the directory that you run the code,

file name	description
C2H2_pulse.data	dipole moment as functions of energy
C2H2_rt.data	components of change of dipole moment (electrons/plus definition) and total dipole moment (electrons/minus + ions/plus) as functions of time
C2H2_rt_energy.data	components of total energy and difference of total energy as functions of time
PS_C_KY_n.dat	information on pseodupotential file for carbon atom
PS_H_KY_n.dat	information on pseodupotential file for hydrogen atom

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/03_C2H2_rt.zip

Explanations of the files are described below:

C2H2_pulse.data

Time-frequency Fourier transformation of the dipole moment.

```
# Fourier-transform spectra:
# energy: Frequency
# dm: Dopile moment
# 1:energy[eV] 2:Re(dm_x)[fs*Angstrom] 3:Re(dm_y)[fs*Angstrom] 4:Re(dm_
↪z)[fs*Angstrom] 5:Im(dm_x)[fs*Angstrom] 6:Im(dm_y)[fs*Angstrom] 7:Im(dm_
↪z)[fs*Angstrom] 8:|dm_x|^2[fs^2*Angstrom^2] 9:|dm_y|^2[fs^2*Angstrom^2] 10:|dm_z|^
↪2[fs^2*Angstrom^2]
```

C2H2_rt.data

Results of time evolution calculation for vector potential, electric field, and dipole moment.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# ddm_e: Change of dipole moment (electrons/plus definition)
# dm: Total dipole moment (electrons/minus + ions/plus)
# 1:Time[fs] 2:Ac_ext_x[fs*V/Angstrom] 3:Ac_ext_y[fs*V/Angstrom] 4:Ac_ext_z[fs*V/
↪Angstrom] 5:E_ext_x[V/Angstrom] 6:E_ext_y[V/Angstrom] 7:E_ext_z[V/Angstrom] 8:Ac_
↪tot_x[fs*V/Angstrom] 9:Ac_tot_y[fs*V/Angstrom] 10:Ac_tot_z[fs*V/Angstrom] 11:E_tot_
↪x[V/Angstrom] 12:E_tot_y[V/Angstrom] 13:E_tot_z[V/Angstrom] 14:ddm_e_x[Angstrom]␣
↪15:ddm_e_y[Angstrom] 16:ddm_e_z[Angstrom] 17:dm_x[Angstrom] 18:dm_y[Angstrom] 19:dm_
↪z[Angstrom]
```

C2H2_rt_energy.data

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
# Eall0: Initial energy
# 1:Time[fs] 2:Eall[eV] 3:Eall-Eall0[eV]
```

3.3 Crystalline silicon (periodic solids)

3.3.1 Exercise-4: Ground state of crystalline silicon

In this exercise, we learn the the ground state calculation of the crystalline silicon of a diamond structure. Calculation is done in a cubic unit cell that contains eight silicon atoms. This exercise will be useful to learn how to set up calculations in SALMON for any periodic systems such as crystalline solid.

Input files

To run the code, following files in samples are used:

file name	description
Si_gs.inp	input file that contains input keywords and their values
Si_rps.dat	pseodupotential file for silicon atom

In the input file *Si_gs.inp*, input keywords are specified. Most of them are mandatory to execute the ground state calculation. This will help you to prepare an input file for other systems that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 04: Ground state of crystalline silicon(periodic solids)
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').
↪    !
!    The manual can be obtained from: https://salmon-tddft.jp/documents.html
↪    !
! * Input format consists of group of keywords like:
↪    !
!      &group
↪    !
!         input keyword = xxx
↪    !
!      /
↪    !
!    (see chapter: 'List of all input keywords' in the manual)
↪    !
!################################################################################
↪###!

&calculation
  !type of theory
  theory = 'dft'
/
```

(continues on next page)

(continued from previous page)

```
&control
  !common name of output files
  sysname = 'Si'
/

&units
  !units used in input and output files
  unit_system = 'a.u.'
/

&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !----------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/

&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/

&scf
  !maximum number of scf iteration and threshold of convergence
  nscf      = 300
```

(continues on next page)

(continued from previous page)

```
  threshold = 1.0d-9
/

&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

We present their explanations below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'dft'
/
```

This indicates that the ground state calculation by DFT is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'Si'
/
```

'Si' defined by `sysname = 'Si'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'a.u.'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/
```

`yn_periodic = 'y'` indicates that three dimensional periodic boundary condition (bulk crystal) is assumed. `al(1:3) = 10.26d0, 10.26d0, 10.26d0` specifies the lattice constans of the unit cell. `nelem = 1` and `natom = 8` indicate the number of elements and the number of atoms in the system, respectively. `nelec = 32` indicate the number of valence electrons in the system. `nstate = 32` indicates the number of Kohn-Sham orbitals to be solved. See *&system in Inputs* for more information.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !-----------------------------------------------------!
/
```

`file_pseudo(1) = './Si_rps.dat'` indicates the pseudopotential filename of element. `izatom(1) = 14` indicates the atomic number of the element. `lloc_ps(1) = 2` indicate the angular momentum of the pseudopotential that will be treated as local.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !number of spatial grids(x,y,z)
```

(continues on next page)

(continued from previous page)

```
  num_rgrid(1:3) = 12, 12, 12
/
```

`num_rgrid(1:3) = 12, 12, 12` specifies the number of the grids for each Cartesian direction. See *&rgrid in Inputs* for more information.

&rgrid

Mandatory: none

```
&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/
```

This input keyword provides grid spacing of k-space for periodic systems.

&scf

Mandatory: nscf, threshold

```
&scf
  !maximum number of scf iteration and threshold of convergence
  nscf      = 300
  threshold = 1.0d-9
/
```

`nscf` is the number of scf iterations. The scf loop in the ground state calculation ends before the number of the scf iterations reaches `nscf`, if a convergence criterion is satisfied. `threshold = 1.0d-9` indicates threshold of the convergence for scf iterations.

&atomic_coor

Mandatory: atomic_coor or atomic_red_coor (it may be provided as a separate file)

```
&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

Cartesian coordinates of atoms are specified in a reduced coordinate system. First column indicates the element, next three columns specify reduced Cartesian coordinates of the atoms, and the last column labels the element.

Output files

After the calculation, following output files and a directory are created in the directory that you run the code,

name	description
Si_info.data	information on ground state solution
Si_eigen.data	energy eigenvalues of orbitals
Si_k.data	k-point distribution
PS_Si_KY_n.dat	information on pseodupotential file for silicon atom
data_for_restart	directory where files used in the real-time calculation are contained

You may download the above files (zipped file, except for the directory *data_for_restart*) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/04_bulkSi_gs.zip
(zipped output files)

Main results of the calculation such as orbital energies are included in *Si_info.data*. Explanations of the *Si_info.data* and other output files are below:

Si_info.data

Calculated orbital and total energies as well as parameters specified in the input file are shown in this file.

Si_eigen.data

1 particle energies.

```
#esp: single-particle energies (eigen energies)
#occ: occupation numbers, io: orbital index
# 1:io, 2:esp[a.u.], 3:occ
```

Si_k.data

k-point distribution.

```
# ik: k-point index
# kx,ky,kz: Reduced coordinate of k-points
# wk: Weight of k-point
# 1:ik[none] 2:kx[none] 3:ky[none] 4:kz[none] 5:wk[none]
# coefficients (2*pi/a [a.u.]) in kx, ky, kz
```

3.3.2 Exercise-5: Dielectric function of crystalline silicon

In this exercise, we learn the linear response calculation of the crystalline silicon of a diamond structure. Calculation is done in a cubic unit cell that contains eight silicon atoms. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*. An impulsive perturbation is applied to all electrons in the unit cell along z direction. Since the dielectric function is isotropic in the diamond structure, calculated dielectric function should not depend on the direction of the perturbation. During the time evolution, electric current averaged over the unit cell volume is calculated. A time-frequency Fourier transformation of the electric current gives us a frequency-dependent conductivity. The dielectric function may be obtained from the conductivity using a standard relation.

Input files

To run the code, following files in samples are used:

You may download the above files (zipped file, except for *restart*) from:
https://salmon-tddft.jp/webmanual/v_1_2_0/exercise_zip_files/Si_gs_rt_response_input.zip

In the input file *Si_rt_response.inp*, input keywords are specified. Most of them are mandatory to execute the calculation. This will help you to prepare the input file for other systems that you want to calculate. A complete list of the input keywords can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 05: Dielectric function of crystalline silicon                      ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html     ␣
↪    !
! * Input format consists of group of keywords like:                            ␣
↪    !
!     &group                                                                    ␣
↪    !
!       input keyword = xxx                                                     ␣
↪    !
!     /                                                                         ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                   ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * Copy the ground state data directory('data_for_restart') (or make symbolic link) ␣
↪    !
!   calculated in 'samples/exercise_04_bulkSi_gs/' and rename the directory to
↪'restart/'!
!   in the current directory.                                                   ␣
↪    !
!################################################################################
↪###!

&calculation
  !type of theory
  theory = 'tddft_response'
/

&control
  !common name of output files
  sysname = 'Si'
/

&units
  !units used in input and output files
  unit_system = 'a.u.'
/

&system
  !periodic boundary condition
  yn_periodic = 'y'
```

(continues on next page)

(continued from previous page)

```
  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution -------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !-----------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/

&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/

&emfield
  !envelope shape of the incident pulse('impulse': impulsive field)
  ae_shape1 = 'impulse'

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !-------------------------------------------------------------------!
/
```

(continues on next page)

(continued from previous page)

```
&analysis
  !energy grid size and number of energy grids for output files
  de      = 1.0d-2
  nenergy = 5000
/

&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !------------------------------------------------------------!
/
```

We present explanations of the input keywords that appear in the input file below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'tddft_response'
/
```

This indicates that the real time (RT) calculation to obtain response function is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'Si'
/
```

'Si' defined by `sysname = 'Si'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'a.u.'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, state, nelem, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&system in Exercise-4*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !-----------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&pseudo in Exercise-4*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/
```

`num_rgrid(1:3) = 12, 12, 12` specifies the number of the grids for each Cartesian direction. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&kgrid

Mandatory: none

```
&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/
```

This input keyword provides grid spacing of k-space for periodic systems. This must be the same as that in the ground state calculation.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/
```

`dt = 0.08d0` specifies the time step of the time evolution calculation. `nt = 6000` specifies the number of time steps in the calculation.

&emfield

Mandatory:ae_shape1

```
&emfield
  !envelope shape of the incident pulse('impulse': impulsive field)
  ae_shape1 = 'impulse'

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

`as_shape1 = 'impulse'` indicates that a weak impulsive field is applied to all electrons at *t=0* `epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0` specify a unit vector that indicates the direction of the impulse. See *&emfield in Inputs* for detail.

&analysis

Mandatory: none

```
&analysis
  !energy grid size and number of energy grids for output files
  de      = 1.0d-2
  nenergy = 5000
/
```

`de = 1.0d-2` specifies the energy spacing in the time-frequency Fourier transformation. `nenergy = 5000` specifies the number of energy steps, and

&atomic_red_coor

Mandatory: atomic_coor or atomic_red_coor (they may be provided as a separate file)

```
&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

Cartesian coordinates of atoms are specified in a reduced coordinate system. First column indicates the element, next three columns specify reduced Cartesian coordinates of the atoms, and the last column labels the element.

Output files

After the calculation, following output files are created in the directory that you run the code,

file name	description
Si_response.data	Fourier spectra of the conductivity and dielectric functions
Si_rt.data	vector potential, electric field, and matter current as functions of time
Si_rt_energy	components of total energy and difference of total energy as functions of time
PS_Si_KY_n.dat	information on pseodupotential file for silicon atom

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/05_bulkSi_lr.zip

Explanations of the output files are described below:

Si_response.data

Time-frequency Fourier transformation of the macroscopic current gives the conductivity of the system. Then the dielectric function is calculated.

```
# Fourier-transform spectra:
# sigma: Conductivity
# eps: Dielectric constant
# 1:Energy[a.u.] 2:Re(sigma_x)[a.u.] 3:Re(sigma_y)[a.u.] 4:Re(sigma_z)[a.u.]␣
↪5:Im(sigma_x)[a.u.] 6:Im(sigma_y)[a.u.] 7:Im(sigma_z)[a.u.] 8:Re(eps_x)[none]␣
↪9:Re(eps_y)[none] 10:Re(eps_z)[none] 11:Im(eps_x)[none] 12:Im(eps_y)[none]␣
↪13:Im(eps_z)[none]
```

Si_rt.data

Results of time evolution calculation for vector potential, electric field, and matter current density.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# Jm: Matter current density (electrons)
# 1:Time[a.u.] 2:Ac_ext_x[a.u.] 3:Ac_ext_y[a.u.] 4:Ac_ext_z[a.u.] 5:E_ext_x[a.u.] 6:E_
↪ext_y[a.u.] 7:E_ext_z[a.u.] 8:Ac_tot_x[a.u.] 9:Ac_tot_y[a.u.] 10:Ac_tot_z[a.u.]␣
↪11:E_tot_x[a.u.] 12:E_tot_y[a.u.] 13:E_tot_z[a.u.]  14:Jm_x[a.u.] 15:Jm_y[a.u.]␣
↪16:Jm_z[a.u.]
```

Si_rt_energy

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
# Eall0: Initial energy
# 1:Time[a.u.] 2:Eall[a.u.] 3:Eall-Eall0[a.u.]
```

3.3.3 Exercise-6: Electron dynamics in crystalline silicon under a pulsed electric field

In this exercise, we learn the calculation of electron dynamics in a unit cell of crystalline silicon of a diamond structure. Calculation is done in a cubic unit cell that contains eight silicon atoms. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*. A pulsed electric field that has cos^2 envelope shape is applied. The parameters that characterize the pulsed field such as magnitude, frequency, polarization, and carrier envelope phase are specified in the input file.

Input files

To run the code, following files in samples are used:

file name	description
Si_rt_pulse.inp	input file that contain input keywords and their values.
Si_rps.dat	pseodupotential file for Carbon
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

You may download the above 2 files (zipped file, except for *restart*) from:
https://salmon-tddft.jp/webmanual/v_1_2_0/exercise_zip_files/Si_gs_rt_pulse_input.zip

In the input file *Si_rt_pulse.inp*, input keywords are specified. Most of them are mandatory to execute the calculation. This will help you to prepare the input file for other systems that you want to calculate. A complete list of the input keywords can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 06: Electron dynamics in crystalline silicon under a pulsed electric␣
↪field   !
```

(continues on next page)

(continued from previous page)

```
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪   !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html      ␣
↪   !
! * Input format consists of group of keywords like:                            ␣
↪   !
!     &group                                                                    ␣
↪   !
!       input keyword = xxx                                                     ␣
↪   !
!     /                                                                         ␣
↪   !
!   (see chapter: 'List of all input keywords' in the manual)                   ␣
↪   !
!-------------------------------------------------------------------------------
↪---!
! * Copy the ground state data directory('data_for_restart') (or make symbolic link) ␣
↪   !
!   calculated in 'samples/exercise_04_bulkSi_gs/' and rename the directory to
↪'restart/'!
!   in the current directory.                                                   ␣
↪   !
!###############################################################################
↪###!

&calculation
  !type of theory
  theory = 'tddft_pulse'
/

&control
  !common name of output files
  sysname = 'Si'
/

&units
  !units used in input and output files
  unit_system = 'a.u.'
/

&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/

&pseudo
```

(continues on next page)

(continued from previous page)

```
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !----------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/

&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/

&emfield
  !envelope shape of the incident pulse('Acos2': cos^2 type envelope for vector␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 5.0d11

  !duration of the incident pulse
  tw1 = 441.195136248d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 0.05696145187d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

(continues on next page)

(continued from previous page)

```
&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !------------------------------------------------------------!
/
```

We present explanations of the input keywords that appear in the input file below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'tddft_response'
/
```

This indicates that the real time (RT) calculation to obtain response function is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'Si'
/
```

'Si' defined by `sysname = 'Si'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'a.u.'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, state, nelem, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&system in Exercise-4*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !----------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&pseudo in Exercise-4*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/
```

`num_rgrid(1:3) = 12, 12, 12` specifies the number of the grids for each Cartesian direction. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&kgrid

Mandatory: none

```
&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/
```

This input keyword provides grid spacing of k-space for periodic systems. This must be the same as that in the ground state calculation.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/
```

`dt = 0.08d0` specifies the time step of the time evolution calculation. `nt = 6000` specifies the number of time steps in the calculation.

&emfield

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  !envelope shape of the incident pulse('Acos2': cos^2 type envelope for vector␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 5.0d11

  !duration of the incident pulse
  tw1 = 441.195136248d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 0.05696145187d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution -----------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                 !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !-------------------------------------------------------------------!
/
```

These input keywords specify the pulsed electric field applied to the system.

`ae_shape1 = 'Acos2'` specifies the envelope of the pulsed electric field, cos^2 envelope for the vector potential.

`I_wcm2_1 = 5.0d11` specifies the maximum intensity of the applied electric field in unit of W/cm^2.

`tw1 = 441.195136248d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 0.05696145187d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0` specify the real part of the unit polarization vector of the pulsed electric field. Specifying only the real part, it describes a linearly polarized pulse.

See *&emfield in Inputs* for detail.

&atomic_red_coor

Mandatory: atomic_coor or atomic_red_coor (they may be provided as a separate file)

```
&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

Cartesian coordinates of atoms are specified in a reduced coordinate system. First column indicates the element, next three columns specify reduced Cartesian coordinates of the atoms, and the last column labels the element.

Output files

After the calculation, following output files are created in the directory that you run the code,

file name	description
Si_pulse.data	matter current and electric field as functions of energy
Si_rt.data	vector potential, electric field, and matter current as functions of time
Si_rt_energy	components of total energy and difference of total energy as functions of time
PS_Si_KY_n.dat	information on pseodupotential file for silicon atom

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/06_bulkSi_rt.zip

Explanations of the output files are described below:

Si_pulse.data

Time-frequency Fourier transformation of the matter current and electric field.

```
# Fourier-transform spectra:
# energy: Frequency
# Jm: Matter current
# E_ext: External electric field
# E_tot: Total electric field
# 1:energy[a.u.] 2:Re(Jm_x)[a.u.] 3:Re(Jm_y)[a.u.] 4:Re(Jm_z)[a.u.] 5:Im(Jm_x)[a.u.]␣
↪6:Im(Jm_y)[a.u.] 7:Im(Jm_z)[a.u.] 8:|Jm_x|^2[a.u.] 9:|Jm_y|^2[a.u.] 10:|Jm_z|^2[a.u.
↪] 11:Re(E_ext_x)[a.u.] 12:Re(E_ext_y)[a.u.] 13:Re(E_ext_z)[a.u.] 14:Im(E_ext_x)[a.u.
↪] 15:Im(E_ext_y)[a.u.] 16:Im(E_ext_z)[a.u.] 17:|E_ext_x|^2[a.u.] 18:|E_ext_y|^2[a.u.
↪] 19:|E_ext_z|^2[a.u.] 20:Re(E_ext_x)[a.u.] 21:Re(E_ext_y)[a.u.] 22:Re(E_ext_z)[a.u.
↪] 23:Im(E_ext_x)[a.u.] 24:Im(E_ext_y)[a.u.] 25:Im(E_ext_z)[a.u.] 26:|E_ext_x|^2[a.u.
↪] 27:|E_ext_y|^2[a.u.] 28:|E_ext_z|^2[a.u.]
```

(continues on next page)

(continued from previous page)

Si_rt.data

Results of time evolution calculation for vector potential, electric field, and matter current density.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# Jm: Matter current density (electrons)
# 1:Time[a.u.] 2:Ac_ext_x[a.u.] 3:Ac_ext_y[a.u.] 4:Ac_ext_z[a.u.] 5:E_ext_x[a.u.] 6:E_
↪ext_y[a.u.] 7:E_ext_z[a.u.] 8:Ac_tot_x[a.u.] 9:Ac_tot_y[a.u.] 10:Ac_tot_z[a.u.]␣
↪11:E_tot_x[a.u.] 12:E_tot_y[a.u.] 13:E_tot_z[a.u.]  14:Jm_x[a.u.] 15:Jm_y[a.u.]␣
↪16:Jm_z[a.u.]
```

Si_rt_energy

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
# Eall0: Initial energy
# 1:Time[a.u.] 2:Eall[a.u.] 3:Eall-Eall0[a.u.]
```

3.4 Maxwell + TDDFT multiscale simulation

3.4.1 Exercise-7: Pulsed-light propagation through a silicon thin film

In this exercise, we learn the calculation of the propagation of a pulsed light through a thin film of crystalline silicon. We consider a silicon thin film of 42 nm thickness, and an irradiation of a few-cycle, linearly polarized pulsed light normally on the thin film. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*. The pulsed light locates in the vacuum region in front of the thin film. The parameters that characterize the pulsed light such as magnitude and frequency are specified in the input file.

Input files

To run the code, following files in samples are used:

file name	description
Si_rt_multiscale.inp	input file that contain input keywords and their values.
Si_rps.dat	pseodupotential file for silicon
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

You may download the above two files (zipped file, except for *restart*) from:
https://salmon-tddft.jp/webmanual/v_1_2_0/exercise_zip_files/Si_gs_rt_multiscale_input.zip

In the input file *Si_rt_multiscale.inp*, input keywords are specified. Most of them are mandatory to execute the calculation. This will help you to prepare the input file for other systems that you want to calculate. A complete list of the input keywords can be found in *List of all input keywords*.

```
!#############################################################################################
↪###!
! Excercise 07: Maxwell+TDDFT multiscale simulation                                          ␣
↪    !
!               (Pulsed-light propagation through a silicon thin film)                       ␣
↪    !
!---------------------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').       ␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html                  ␣
↪    !
! * Input format consists of group of keywords like:                                         ␣
↪    !
!     &group                                                                                 ␣
↪    !
!       input keyword = xxx                                                                  ␣
↪    !
!     /                                                                                      ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                                ␣
↪    !
!---------------------------------------------------------------------------------------------
↪---!
! * Copy the ground state data directory('data_for_restart') (or make symbolic link)        ␣
↪    !
!   calculated in 'samples/exercise_04_bulkSi_gs/' and rename the directory to
↪'restart/'!
!   in the current directory.                                                                ␣
↪    !
!#############################################################################################
↪###!

&calculation
  !type of theory
  theory = 'multi_scale_maxwell_tddft'
/

&control
  !common name of output files
  sysname = 'Si'
/

&units
  !units used in input and output files
  unit_system = 'a.u.'
/

&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0
```

(continues on next page)

(continued from previous page)

```
  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !----------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/

&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/

&emfield
  !envelope shape of the incident pulse('Acos2': cos^2 type envelope for vector␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.0d12

  !duration of the incident pulse
  tw1 = 441.195136248d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 0.05696145187d0
```

(continues on next page)

(continued from previous page)

```
  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/

&multiscale
  !number of macro grids in electromagnetic analysis for x, y, and z directions
  nx_m = 8
  ny_m = 1
  nz_m = 1

  !macro grid spacing for x, y, and z directions
  hx_m = 100.0d0
  hy_m = 100.0d0
  hz_m = 100.0d0

  !number of macroscopic grids for vacumm region
  !(nxvacl_m is for negative x-direction in front of material)
  !(nxvacr_m is for positive x-direction behind material)
  nxvacl_m = 1000
  nxvacr_m = 1000
/

&maxwell
  !boundary condition of electromagnetic analysis
  !first index(1-3 rows) corresponds to x, y, and z directions
  !second index(1-2 columns) corresponds to bottom and top of the directions
  !('abc' is absorbing boundary condition)
  boundary_em(1,1) = 'abc'
  boundary_em(1,2) = 'abc'
/

&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'      .0      .0      .0      1
  'Si'      .25     .25     .25     1
  'Si'      .5      .0      .5      1
  'Si'      .0      .5      .5      1
  'Si'      .5      .5      .0      1
  'Si'      .75     .25     .75     1
  'Si'      .25     .75     .75     1
  'Si'      .75     .75     .25     1
  !--- Format ------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-----------------------------------------------------------!
/
```

We present explanations of the input keywords that appear in the input file below:

Required and recommened variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'multi_scale_maxwell_tddft'
/
```

This indicates that the multi-scale Maxwell-TDDFT calculation is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'Si'
/
```

'Si' defined by `sysname = 'Si'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'a.u.'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, state, nelem, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'y'

  !grid box size(x,y,z)
  al(1:3) = 10.26d0, 10.26d0, 10.26d0

  !number of elements, atoms, electrons and states(bands)
  nelem  = 1
  natom  = 8
  nelec  = 32
  nstate = 32
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&system in Exercise-4*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './Si_rps.dat'

```

(continues on next page)

(continued from previous page)

```
  !atomic number of element
  izatom(1) = 14

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 2
  !--- Caution ------------------------------------------!
  ! Index must correspond to those in &atomic_red_coor.   !
  !-----------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the ground state calculation. See *&pseudo in Exercise-4*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 12, 12, 12
/
```

`num_rgrid(1:3) = 12, 12, 12` specifies the number of the grids for each Cartesian direction. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&kgrid

Mandatory: none

```
&kgrid
  !number of k-points(x,y,z)
  num_kgrid(1:3) = 4, 4, 4
/
```

This input keyword provides grid spacing of k-space for periodic systems. This must be the same as that in the ground state calculation.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.08d0
  nt = 6000
/
```

`dt = 0.08d0` specifies the time step of the time evolution calculation. `nt = 6000` specifies the number of time steps in the calculation.

&emfield

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  !envelope shape of the incident pulse('Acos2': cos^2 type envelope for vector␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.0d12

  !duration of the incident pulse
  tw1 = 441.195136248d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 0.05696145187d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

These input keywords specify the pulsed electric field applied to the system.

`ae_shape1 = 'Acos2'` specifies the envelope of the pulsed electric field, cos^2 envelope for the vector potential.

`I_wcm2_1 = 1.0d12` specifies the maximum intensity of the applied electric field in unit of W/cm^2.

`tw1 = 441.195136248d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 0.05696145187d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0` specify the real part of the unit polarization vector of the pulsed electric field. Specifying only the real part, it describes a linearly polarized pulse.

See *&emfield in Inputs* for detail.

&multiscale

```
&multiscale
  !number of macro grids in electromagnetic analysis for x, y, and z directions
  nx_m = 8
  ny_m = 1
  nz_m = 1

  !macro grid spacing for x, y, and z directions
  hx_m = 100.0d0
  hy_m = 100.0d0
  hz_m = 100.0d0

  !number of macroscopic grids for vacumm region
  !(nxvacl_m is for negative x-direction in front of material)
  !(nxvacr_m is for positive x-direction behind material)
  nxvacl_m = 1000
```

(continues on next page)

(continued from previous page)

```
  nxvacr_m = 1000
/
```

This input keyword specifies information necessary for Maxwell-TDDFT multiscale calculations.

`nx_m = 8` specifies the number of the macroscopic grid points for x-direction in the spatial region where the material exists. `ny_m = 1` and `nz_m = 1` are those for y- and z-directions.

`hx_m = 100.0d0` specifies the grid spacing of the macroscopic grid for x-direction. `hy_m = 100.0d0` and `hz_m = 100.0d0` are those for y- and z-directions.

`nxvacl_m = 1000` and `nxvacr_m = 1000` indicate the number of grid points in the vacuum region, `nxvacl_m` for the left and `nxvacr_m` for the right from the surface of the material.

&maxwell

```
&maxwell
  !boundary condition of electromagnetic analysis
  !first index(1-3 rows) corresponds to x, y, and z directions
  !second index(1-2 columns) corresponds to bottom and top of the directions
  !('abc' is absorbing boundary condition)
  boundary_em(1,1) = 'abc'
  boundary_em(1,2) = 'abc'
/
```

`boundary_em(1,1) = 'abc'` and boundary_em(1,2) = 'abc' set the abosorbing bondary conditions in electromagnetic analysis. The first index(1-3 rows) corresponds to x, y, and z axes. The second index(1-2 columns) corresponds to bottom and top of the axes.

&atomic_red_coor

Mandatory: atomic_coor or atomic_red_coor (they may be provided as a separate file)

```
&atomic_red_coor
  !cartesian atomic reduced coodinates
  'Si'       .0      .0      .0      1
  'Si'       .25     .25     .25     1
  'Si'       .5      .0      .5      1
  'Si'       .0      .5      .5      1
  'Si'       .5      .5      .0      1
  'Si'       .75     .25     .75     1
  'Si'       .25     .75     .75     1
  'Si'       .75     .75     .25     1
  !--- Format ---------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) !
  !-------------------------------------------------------------!
/
```

Cartesian coordinates of atoms are specified in a reduced coordinate system. First column indicates the element, next three columns specify reduced Cartesian coordinates of the atoms, and the last column labels the element.

Output files

After the calculation, new directory *multiscale/* is created, then, following output files are created in the directory,

file name	description
Si_m/mxxxxxx/Si_rt.data	vector potential, electric field, and matter current at macroscopic position *xxxxxx* as functions of time
Si_m/mxxxxxx/Si_rt_energy.data	components of total energy and difference of total energy at macroscopic position *xxxxxx* as functions of time
Si_m/mxxxxxx/PS_Si_KY_rinf	information on pseodupotential file for silicon atom at macroscopic position *xxxxxx*
Si_RT_Ac/Si_Ac_yyyyyy.data	vector potential, electric field, magnetic field, electromagnetic current density at time step *yyyyyy* as function of space
Si_wave.data	amplitudes of incident, reflected, and transmitted wave

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/07_bulkSi_ms.zip

Explanations of the output files are described below:

Si_m/mxxxxxx/Si_rt.data

The number in the file name specifies the macroscopic position. Results of time evolution calculation for vector potential, electric field, and matter current density.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# Jm: Matter current density (electrons)
# 1:Time[a.u.] 2:Ac_ext_x[a.u.] 3:Ac_ext_y[a.u.] 4:Ac_ext_z[a.u.] 5:E_ext_x[a.u.] 6:E_
↪ext_y[a.u.] 7:E_ext_z[a.u.] 8:Ac_tot_x[a.u.] 9:Ac_tot_y[a.u.] 10:Ac_tot_z[a.u.]␣
↪11:E_tot_x[a.u.] 12:E_tot_y[a.u.] 13:E_tot_z[a.u.]  14:Jm_x[a.u.] 15:Jm_y[a.u.]␣
↪16:Jm_z[a.u.]
```

Si_m/mxxxxxx/Si_rt_energy.data

The number in the file name specifies the macroscopic position. *Eall* and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
   # Eall0: Initial energy
```

1:Time[a.u.] 2:Eall[a.u.] 3:Eall-Eall0[a.u.]

Si_RT_Ac/Si_Ac_yyyyyy.data

The number in the file name specifies the iteration number. Various quantities at a time are shown as function of macroscopic position.

```
# Multiscale TDDFT calculation
# IX, IY, IZ: FDTD Grid index
# x, y, z: Coordinates
# Ac: Vector potential field
# E: Electric field
# J_em: Electromagnetic current density
# 1:IX[none] 2:IY[none] 3:IZ[none] 4:Ac_x[a.u.] 5:Ac_y[a.u.] 6:Ac_z[a.u.] 7:E_x[a.u.]␣
↪8:E_y[a.u.] 9:E_z[a.u.] 10:B_x[a.u.] 11:B_y[a.u.] 12:B_z[a.u.] 13:Jem_x[a.u.]␣
↪14:Jem_y[a.u.] 15:Jem_z[a.u.] 16:E_em[a.u./vol] 17:E_abs[a.u./vol]
```

(continues on next page)

(continued from previous page)

Si_wave.data

Amplitudes of incident, reflected, and transmitted wave.

```
# 1D multiscale calculation:
# E_inc: E-field amplitude of incident wave
# E_ref: E-field amplitude of reflected wave
# E_tra: E-field amplitude of transmitted wave
# 1:Time[a.u.] 2:E_inc_x[a.u.] 3:E_inc_y[a.u.] 4:E_inc_z[a.u.] 5:E_ref_x[a.u.] 6:E_
↪ref_y[a.u.] 7:E_ref_z[a.u.] 8:E_tra_x[a.u.] 9:E_tra_y[a.u.] 10:E_tra_z[a.u.]
```

3.5 Geometry optimization and Ehrenfest molecular dynamics

3.5.1 Exercise-8: Geometry optimization of C2H2 molecule

In this exercise, we learn the calculation of geometry optimization of acetylene (C2H2) molecule, solving the static Kohn-Sham equation. This exercise will be useful to learn how to set up calculations in SALMON for any isolated systems such as molecules and nanoparticles.

Input files

To run the code, following files in samples are used:

file name	description
C2H2_opt.inp	input file that contains input keywords and their values
C_rps.dat	pseodupotential file for carbon atom
H_rps.dat	pseudopotential file for hydrogen atom

You may download the above 3 files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_1_2_0/exercise_zip_files/C2H2_gs_input.zip
(zipped input and pseudopotential files)

In the input file *C2H2_opt.inp*, input keywords are specified. Most of them are mandatory to execute the geometry optimization. This will help you to prepare an input file for other systems that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 08: Geometry optimization of C2H2 molecule                          ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html    ␣
↪    !
! * Input format consists of group of keywords like:                           ␣
↪    !
```

(continues on next page)

(continued from previous page)

```
!     &group                                                                    ␣
↪    !
!        input keyword = xxx                                                   ␣
↪    !
!     /                                                                        ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                  ␣
↪    !
!##############################################################################
↪###!

&calculation
  !type of theory
  theory = 'dft'

  !geometry optimization option
  yn_opt = 'y'
/

&control
  !common name of output files
  sysname = 'C2H2'
/

&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/

&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 12.0d0, 12.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
```

(continues on next page)

(continued from previous page)

```
  !-------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.20d0, 0.20d, 0.20d0
/

&scf
  !maximum number of scf iteration and threshold of convergence for ground state␣
↪calculation
  nscf      = 300
  threshold = 1.0d-9
/

&opt
  !threshold(maximum force on atom) of convergence for geometry optimization
  convrg_opt_fmax = 1.0d-3
/

&atomic_coor
  !cartesian atomic coodinates
  'C'    0.0    0.0    0.6  1  y
  'H'    0.0    0.0    1.7  2  y
  'C'    0.0    0.0   -0.6  1  y
  'H'    0.0    0.0   -1.7  2  y
  !--- Format ------------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) y/n !
  !--- Caution -----------------------------------------------------!
  ! final index(y/n) determines free/fix for the atom coordinate.    !
  !-------------------------------------------------------------------!
/
```

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'dft'

  !geometry optimization option
  yn_opt = 'y'
/
```

`theory = 'dft'` indicates that the ground state calculation by DFT is carried out in the present job. See *&calculation in Inputs* for detail. `yn_opt = 'y'` indicates that the geometry optimization calculation is performed.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, nelem, natom, nelec, nstate

```
&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 12.0d0, 12.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

`yn_periodic = 'n'` indicates that the isolated boundary condition will be used in the calculation. `al(1:3) = 12.0d0, 12.0d0, 16.0d0` specifies the lengths of three sides of the rectangular parallelepiped where the grid points are prepared. `nelem = 2` and `natom = 4` indicate the number of elements and the number of atoms in the system, respectively. `nelec = 10` indicate the number of valence electrons in the system. `nstate = 6` indicates the number of Kohn-Sham orbitals to be solved. Since the present code assumes that the system is spin saturated, `nstate` should be equal to or larger than `nelec/2`. See *&system in Inputs* for more information.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
```

(continues on next page)

(continued from previous page)

```
  lloc_ps(2) = 0
  !--- Caution ---------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !-------------------------------------------------!
/
```

Parameters related to atomic species and pseudopotentials. `file_pseudo(1) = './C_rps.dat'` indicates the filename of the pseudopotential of element. `izatom(1) = 6` specifies the atomic number of the element. `lloc_ps(1) = 1` specifies the angular momentum of the pseudopotential that will be treated as local.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.20d0, 0.20d0, 0.20d0
/
```

`dl(1:3) = 0.20d0, 0.20d0, 0.20d0` specifies the grid spacings in three Cartesian directions. See *&rgrid in Inputs* for more information.

&scf

Mandatory: nscf, threshold

```
&scf
  !maximum number of scf iteration and threshold of convergence
  nscf      = 300
  threshold = 1.0d-9
/
```

`nscf` is the number of scf iterations. The scf loop in the ground state calculation ends before the number of the scf iterations reaches `nscf`, if a convergence criterion is satisfied. `threshold = 1.0d-9` indicates threshold of the convergence for scf iterations.

&opt

Mandatory:

```
&opt
  !threshold(maximum force on atom) of convergence for geometry optimization
  convrg_opt_fmax = 1.0d-3
/
```

&atomic_coor

Mandatory: atomic_coor or atomic_red_coor (it may be provided as a separate file)

```
&atomic_coor
  !cartesian atomic coodinates
  'C'    0.0    0.0    0.6  1  y
  'H'    0.0    0.0    1.7  2  y
  'C'    0.0    0.0   -0.6  1  y
  'H'    0.0    0.0   -1.7  2  y
  !--- Format ------------------------------------------------------!
  ! 'symbol' x y z index(correspond to that of pseudo potential) y/n !
  !--- Caution -----------------------------------------------------!
  ! final index(y/n) determines free/fix for the atom coordinate.    !
  !------------------------------------------------------------------!
/
```

Cartesian coordinates of atoms. The first column indicates the element. Next three columns specify Cartesian coordinates of the atoms. The number in the next column labels the element. The 'y' at the last column indicates to allow to change atomic coordinate during the optimization. ('n' can be used to fix the atomic cooordinate.)

Output files

After the calculation, following output files and a directory are created in the directory that you run the code,

name	description
C2H2_info.data	information on ground state solution
C2H2_eigen.data	1 particle energies
C2H2_trj.xyz	atomic coordinates during the geometry optimization
C2H2_k.data	k-point distribution (for isolated systems, only gamma point is described)
data_for_restart	directory where files used in the real-time calculation are contained
PS_C_KY_n.dat	information on pseodupotential file for carbon atom
PS_H_KY_n.dat	information on pseodupotential file for hydrogen atom

You may download the above files (zipped file, except for the directory *data_for_restart*) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/08_C2H2_opt.zip
(zipped output files)

Main results of the calculation such as orbital energies are included in *C2H2_info.data*. Explanations of the *C2H2_info.data* and other output files are below:

C2H2_info.data

Calculated orbital and total energies as well as parameters specified in the input file are shown in this file.

C2H2_eigen.data

1 particle energies.

```
#esp: single-particle energies (eigen energies)
#occ: occupation numbers, io: orbital index
# 1:io, 2:esp[eV], 3:occ
```

C2H2_trj.xyz

The atomic coordinates during the geometry optimization in xyz format.

C2H2_k.data

k-point distribution(for isolated systems, only gamma point is described).

```
# ik: k-point index
# kx,ky,kz: Reduced coordinate of k-points
# wk: Weight of k-point
# 1:ik[none] 2:kx[none] 3:ky[none] 4:kz[none] 5:wk[none]
# coefficients (2*pi/a [a.u.]) in kx, ky, kz
```

3.5.2 Exercise-9: Ehrenfest molecular dynamics of C2H2 molecule

In this exercise, we learn the calculation of the molecular dynamics in the acetylene (C2H2) molecule under a pulsed electric field, solving the time-dependent Kohn-Sham equation and the Newtonian equation. As outputs of the calculation, time-evolution of the electron density as well as molecular structures and associated quantities such as the electron and ion kinetic energies, the electric dipole moment of the system and temperature as functions of time are calculated.. This tutorial should be carried out after finishing the geometry optimization that was explained in *Exercise-8*. In the calculation, a pulsed electric field that has cos^2 envelope shape is applied. The parameters that characterize the pulsed field such as magnitude, frequency, polarization direction, and carrier envelope phase are specified in the input file.

Input files

To run the code, following files in samples are used. The directory *restart* is created in the ground state calculation as *data_for_restart*. Pseudopotential files are already used in the geometry optimization. Therefore, *C2H2_md.inp* that specifies input keywords and their values for the pulsed electric field and molecular dynamics calculations is the only file that the users need to prepare.

file name	description
C2H2_md.inp	input file that contain input keywords and their values.
C_rps.dat	pseodupotential file for carbon
H_rps.dat	pseudopotential file for hydrogen
restart	directory created in the geometry optimization (rename the directory from *data_for_restart* to *restart*)

In the input file *C2H2_md.inp*, input keywords are specified. Most of them are mandatory to execute the calculation of electron dynamics induced by a pulsed electric field. This will help you to prepare the input file for other systems and other pulsed electric fields with molecular dynamics calculation that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!################################################################################
↪###!
! Excercise 09: Ehrenfest molecular dynamics of C2H2 molecule                   ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html    ␣
↪    !
! * Input format consists of group of keywords like:                           ␣
↪    !
!     &group                                                                   ␣
↪    !
!       input keyword = xxx                                                    ␣
↪    !
```

(continues on next page)

(continued from previous page)

```
!     /                                                                                 ␣
↪    !
!    (see chapter: 'List of all input keywords' in the manual)                         ␣
↪    !
!-------------------------------------------------------------------------------------
↪---!
! * Ehrenfest-MD option is still trial.                                                ␣
↪    !
! * Copy the ground state data directory ('data_for_restart') (or make symbolic link)␣
↪    !
!    calculated in 'samples/exercise_08_C2H2_opt/' and rename the directory to
↪'restart/' !
!    in the current directory.                                                         ␣
↪    !
!#####################################################################################
↪###!

&calculation
  !type of theory
  theory = 'tddft_pulse'

  !molecular dynamics option
  yn_md  = 'y'
/

&control
  !common name of output files
  sysname = 'C2H2'
/

&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/

&system
  !periodic boundary condition
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 12.0d0, 12.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/

&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1
```

(continues on next page)

(continued from previous page)

```
  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/

&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/

&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.20d0, 0.20d0, 0.20d0
/

&tgrid
  !time step size and number of time grids(steps)
  dt = 1.00d-3
  nt = 5000
/

&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 6.00d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 9.28d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/

&md
  !ensemble
  ensemble = 'NVE'

  !set of initial velocities
```

(continues on next page)

(continued from previous page)

```
  yn_set_ini_velocity = 'y'

  !setting temperature [K] for NVT ensemble, velocity scaling,
  !and generating initial velocities
  temperature0_ion_k = 300.0d0

  !time step interval for updating pseudopotential
  step_update_ps = 20
/
```

We present explanations of the input keywords that appear in the input file below:

required and recommended variables

&calculation

Mandatory: theory

```
&calculation
  !type of theory
  theory = 'tddft_pulse'

  !molecular dynamics option
  yn_md  = 'y'
/
```

This indicates that the real time (RT) calculation for a pulse response is carried out in the present job. See *&calculation in Inputs* for detail. `yn_md = 'y'` indicates that molecular dynamics calculation is coupled with the `theory`, where the Ehrenfest dynamics coupled with the TDDFT is performed in this case.

&control

Mandatory: none

```
&control
  !common name of output files
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files.

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input file. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic, al, nelem, natom, nelectron, nstate

```
&system
  !periodic boundary condition
```

(continues on next page)

(continued from previous page)

```
  yn_periodic = 'n'

  !grid box size(x,y,z)
  al(1:3) = 12.0d0, 12.0d0, 16.0d0

  !number of elements, atoms, electrons and states(orbitals)
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

These input keywords and their values should be the same as those used in the geometry optimization. See *Exercise-8*.

&pseudo

Mandatory: file_pseudo, izatom

```
&pseudo
  !name of input pseudo potential file
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'

  !atomic number of element
  izatom(1) = 6
  izatom(2) = 1

  !angular momentum of pseudopotential that will be treated as local
  lloc_ps(1) = 1
  lloc_ps(2) = 0
  !--- Caution --------------------------------------!
  ! Indices must correspond to those in &atomic_coor. !
  !------------------------------------------------!
/
```

These input keywords and their values should be the same as those used in the geometry optimization. See *Exercise-8*.

&functional

Mandatory: xc

```
&functional
  !functional('PZ' is Perdew-Zunger LDA: Phys. Rev. B 23, 5048 (1981).)
  xc = 'PZ'
/
```

This indicates that the local density approximation with the Perdew-Zunger functional is used.

&rgrid

Mandatory: dl or num_rgrid

```
&rgrid
  !spatial grid spacing(x,y,z)
  dl(1:3) = 0.20d0, 0.20d0, 0.20d0
/
```

`dl(1:3) = 0.20d0, 0.20d0, 0.20d0` specifies the grid spacings in three Cartesian directions. This must be the same as that in the ground state calculation. See *&rgrid in Inputs* for more information.

&tgrid

Mandatory: dt, nt

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 1.00d-3
  nt = 5000
/
```

`dt = 1.00d-3` specifies the time step of the time evolution calculation. `nt = 5000` specifies the number of time steps in the calculation.

&emfield

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 6.00d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 9.28d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !-------------------------------------------------------------------!
/
```

These input keywords specify the pulsed electric field applied to the system.

`ae_shape1 = 'Ecos2'` indicates that the envelope of the pulsed electric field has a *cos*^2 shape.

`I_wcm2_1 = 1.00d8` specifies the maximum intensity of the applied electric field in unit of W/cm^2.

`tw1 = 6.00d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 9.28d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0` specifies the real part of the unit polarization vector of the pulsed electric field. Using the real polarization vector, it describes a linearly polarized pulse.

`phi_cep1 = 0.75d0` specifies the carrier envelope phase of the pulse. As noted above, 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish.

See *&emfield in Inputs* for details.

&md

Mandatory: none

```
&md
  !ensemble
  ensemble = 'NVE'

  !set of initial velocities
  yn_set_ini_velocity = 'y'

  !setting temperature [K] for NVT ensemble, velocity scaling,
  !and generating initial velocities
  temperature0_ion_k = 300.0d0

  !time step interval for updating pseudopotential
  step_update_ps = 20
/
```

These input keywords specify conditions of the molecular dynamics.

`ensemble = 'NVE'` specifies that the microcanonical ensemble is used (thermostat is not used).

`yn_set_ini_velocity = 'y'` indicates that initial velocity is given using random number with the specified temperature by 'temperature0_ion_k'.

`temperature0_ion_k = 300.0d0` specifies the setting temperature for generating the initial velicity (and also for thermostat in NVT ensemble).

`step_update_ps = 20` specifies the time step interval to update pseudopotential.

Output files

After the calculation, following output files are created in the directory that you run the code,

file name	description
C2H2_pulse.data	dipole moment as functions of energy
C2H2_rt.data	components of change of dipole moment (electrons/plus definition) and total dipole moment (electrons/minus + ions/plus) as functions of time
C2H2_rt_energy.data	components of total energy and difference of total energy as functions of time
C2H2_trj.xyz	Trajectory of atoms(ions): Atomic coordinates, velocities, and forces are printed
PS_C_KY_n.dat	information on pseodupotential file for carbon atom
PS_H_KY_n.dat	information on pseodupotential file for hydrogen atom

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/09_C2H2_md.zip

Explanations of the files are described below:

C2H2_pulse.data

Time-frequency Fourier transformation of the dipole moment.

```
# Fourier-transform spectra:
# energy: Frequency
# dm: Dopile moment
# 1:energy[eV] 2:Re(dm_x)[fs*Angstrom] 3:Re(dm_y)[fs*Angstrom] 4:Re(dm_
↪z)[fs*Angstrom] 5:Im(dm_x)[fs*Angstrom] 6:Im(dm_y)[fs*Angstrom] 7:Im(dm_
↪z)[fs*Angstrom] 8:|dm_x|^2[fs^2*Angstrom^2] 9:|dm_y|^2[fs^2*Angstrom^2] 10:|dm_z|^
↪2[fs^2*Angstrom^2]
```

C2H2_rt.data

Results of time evolution calculation for vector potential, electric field, and dipole moment.

```
# Real time calculation:
# Ac_ext: External vector potential field
# E_ext: External electric field
# Ac_tot: Total vector potential field
# E_tot: Total electric field
# ddm_e: Change of dipole moment (electrons/plus definition)
# dm: Total dipole moment (electrons/minus + ions/plus)
# 1:Time[fs] 2:Ac_ext_x[fs*V/Angstrom] 3:Ac_ext_y[fs*V/Angstrom] 4:Ac_ext_z[fs*V/
↪Angstrom] 5:E_ext_x[V/Angstrom] 6:E_ext_y[V/Angstrom] 7:E_ext_z[V/Angstrom] 8:Ac_
↪tot_x[fs*V/Angstrom] 9:Ac_tot_y[fs*V/Angstrom] 10:Ac_tot_z[fs*V/Angstrom] 11:E_tot_
↪x[V/Angstrom] 12:E_tot_y[V/Angstrom] 13:E_tot_z[V/Angstrom] 14:ddm_e_x[Angstrom]␣
↪15:ddm_e_y[Angstrom] 16:ddm_e_z[Angstrom] 17:dm_x[Angstrom] 18:dm_y[Angstrom] 19:dm_
↪z[Angstrom]
```

C2H2_rt_energy.data

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively.

```
# Real time calculation:
# Eall: Total energy
# Eall0: Initial energy
# Tion: Kinetic energy of ions
# Temperature_ion: Temperature of ions
# E_work: Work energy of ions(sum f*dr)
# 1:Time[fs] 2:Eall[eV] 3:Eall-Eall0[eV] # 4:Tion[eV] 5:Temperature_ion[K] 6:E_
↪work[eV]
```

C2H2_trj.xyz

Atomic coordinates [Angstrom], velocities [a.u.] and forces [a.u.] are printed along the time evolution in xyz format.

3.6 FDTD simulation(electromagnetic analysis)

3.6.1 Exercise-10: Pulsed electric field response of a metallic nanosphere in classical electromagnetism(FDTD simulation)

In this exercise, we learn the pulsed electric field response in the metallic nanosphere, solving the time-dependent Maxwell equations. As outputs of the calculation, the time response of the electromagnetic field is calculated. A pulsed electric field that has cos^2 envelope shape is applied. The parameters that characterize the pulsed field such as magnitude, frequency, polarization direction, and carrier envelope phase are specified in the input file.

Input files

To run the code, the input file *classicEM_rt_pulse.inp* that contains input keywords and their values for the pulsed electric field calculation is required. The shape file of the metallic nanosphere *shape.cube* is also required.

The shape file can be generated by program `FDTD_make_shape` in SALMON utilities: https://salmon-tddft.jp/utilities.html

'shape.inp' is an input file for 'FDTD_make_shape' to generate 'shape.cube'.

The input files are in samples

file name	description
classicEM_rt_pulse.inp	input file that contain input keywords and their values.
shape.cube	shape file for fdtd
shape.inp	input file for `FDTD_make_shape`

In the input file *classicEM_rt_pulse.inp*, input keywords are specified. Most of them are mandatory to execute the linear response calculation. This will help you to prepare the input file for other systems that you want to calculate. A complete list of the input keywords that can be used in the input file can be found in *List of all input keywords*.

```
!#########################################################################################
↪###!
! Excercise 10: Pulsed electric field response of a metallic nanosphere                  ␣
↪    !
!               in classical electromagnetism(FDTD simulation)                           ␣
↪    !
!-----------------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is expained in our manual(see chapter: 'Exercises').    ␣
↪    !
!   The manual can be obtained from: https://salmon-tddft.jp/documents.html              ␣
↪    !
! * Input format consists of group of keywords like:                                     ␣
↪    !
!     &group                                                                             ␣
↪    !
!       input keyword = xxx                                                              ␣
↪    !
!     /                                                                                  ␣
↪    !
!   (see chapter: 'List of all input keywords' in the manual)                            ␣
↪    !
!-----------------------------------------------------------------------------------------
↪---!
! * The read-in file 'shape_file' in &maxwell category can be generated by program      ␣
↪    !
!   'FDTD_make_shape' in SALMON utilities(https://salmon-tddft.jp/utilities.html).       ␣
↪    !
!   'shape.inp' is an input file for 'FDTD_make_shape' to generate 'shape.cube'.         ␣
↪    !
! * Results can be visualized by program 'FDTD_make_figani' in SALMON utilities.         ␣
↪    !
!#########################################################################################
↪###!

&calculation
```

(continues on next page)

(continued from previous page)

```
  !type of theory
  theory = 'maxwell'
/

&control
  !name of directory where output files are contained
  base_directory = 'result'
/

&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/

&system
  !periodic boundary condition
  yn_periodic = 'n'
/

&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 4.60d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 5.49d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/

&maxwell
  !box size and spacing of spatial grid(x,y,z)
  al_em(1:3) = 120d0, 120d0, 120d0
  dl_em(1:3) = 1.2d0, 1.2d0, 1.2d0

  !time step size and number of time grids(steps)
  dt_em = 2.30d-4
  nt_em = 20000

  !name of input shape file and number of media in the file
  shape_file = './shape.cube'
```

(continues on next page)

(continued from previous page)

```
  media_num  = 1

  !*** MEDIA INFORMATION(START) ************************************!
  !type of media(media ID)
  media_type(1) = 'lorentz-drude'
  !--- Au described by Lorentz-Drude model ---------------------------!
  ! The parameters are determined from:                               !
  ! (https://www.osapublishing.org/ao/abstract.cfm?uri=ao-37-22-5271) !
  !-------------------------------------------------------------------!

  !number of poles and plasma frequency of media(media ID)
  pole_num_ld(1) = 6
  omega_p_ld(1)  = 9.030d0

  !oscillator strength, collision frequency,
  !and oscillator frequency of media(media ID,pole ID)
  f_ld(1,1:6)     = 0.760d0, 0.024d0, 0.010d0, 0.071d0, 0.601d0, 4.384d0
  gamma_ld(1,1:6) = 0.053d0, 0.241d0, 0.345d0, 0.870d0, 2.494d0, 2.214d0
  omega_ld(1,1:6) = 0.000d0, 0.415d0, 0.830d0, 2.969d0, 4.304d0, 13.32d0
  !*** MEDIA INFORMATION(END) **************************************!

  !*** SOURCE INFORMATION(START) ************************************!
  !type of method to generate the incident pulse
  !('source': incident current source)
  wave_input = 'source'

  !location of source(x,y,z)
  source_loc1(1:3) = -37.8d0, 0.0d0, 0.0d0

  !propagation direction of the incidenty pulse(x,y,z)
  ek_dir1(1:3) = 1.0d0, 0.0d0, 0.0d0
  !*** SOURCE INFORMATION(END) **************************************!

  !*** OBSERVATION INFORMATION(START) *******************************!
  !number of observation points
  obs_num_em = 1

  !time step interval for sampling
  obs_samp_em = 20

  !location of observation point(observation ID,x,y,z)
  obs_loc_em(1,1:3) = 0.0d0, 0.0d0, 0.0d0

  !output flag for electrmagnetic field distribution(observation ID)
  yn_obs_plane_em(1) = 'n'
  !--- Make of animation file -----------------------------------------!
  ! When yn_obs_plane_em(1) = 'y', animation file can be made         !
  ! by program 'FDTD_make_figani' in SALMON utilities.                !
  ! The animation file visualizes electromagnetic field distributions !
  ! on the cross-section including the observation point              !
  ! whose location is determined by obs_loc_em.                       !
  !-------------------------------------------------------------------!
  !*** OBSERVATION END(START) ***************************************!
/
```

We present explanations of the input keywords that appear in the input file below:

required and recommended variables

&calculation

Mandatory: Theory

```
&calculation
  !type of theory
  theory = 'maxwell'
/
```

This indicates that the real time classical electromagnetism calculation is carried out in the present job. See *&calculation in Inputs* for detail.

&control

Mandatory: none

```
&control
  !name of directory where output files are contained
  base_directory = 'result'
/
```

`result` defined by `base_directory = 'result'` will be used in the directory name that contains output files. Default is `directory = './'`

&units

Mandatory: none

```
&units
  !units used in input and output files
  unit_system = 'A_eV_fs'
/
```

This input keyword specifies the unit system to be used in the input and output files. If you do not specify it, atomic unit will be used. See *&units in Inputs* for detail.

&system

Mandatory: yn_periodic

```
&system
  !periodic boundary condition
  yn_periodic = 'n'
/
```

`yn_periodic = 'n'` indicates that the isolated boundary condition will be used in the calculation.

&emfield

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Ecos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d8

  !duration of the incident pulse
  tw1 = 4.60d0
```

(continues on next page)

(continued from previous page)

```
  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 5.49d0

  !polarization unit vector(real part) for the incident pulse(x,y,z)
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0

  !carrier emvelope phase of the incident pulse
  !(phi_cep1 must be 0.25 + 0.5 * n(integer) when ae_shape1 = 'Ecos2')
  phi_cep1 = 0.75d0
  !--- Caution ------------------------------------------------------!
  ! Defenition of the incident pulse is wrriten in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

These input keywords specify the pulsed electric field applied to the system.

`ae_shape1 = 'Ecos2'` indicates that the envelope of the pulsed electric field has a *cos*^2 shape.

`I_wcm2_1 = 1.00d8` specifies the maximum intensity of the applied electric field in unit of W/cm^2.

`tw1 = 4.60d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 5.49d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0` specifies the real part of the unit polarization vector of the pulsed electric field. Using the real polarization vector, it describes a linearly polarized pulse.

`phi_cep1 = 0.75d0` specifies the carrier envelope phase of the pulse. As noted above, 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish.

See *&emfield in Inputs* for details.

&maxwell

Mandatory: al_em, dl_em, nt_em

```
&maxwell
  !box size and spacing of spatial grid(x,y,z)
  al_em(1:3) = 120d0, 120d0, 120d0
  dl_em(1:3) = 1.2d0, 1.2d0, 1.2d0

  !time step size and number of time grids(steps)
  dt_em = 2.30d-4
  nt_em = 20000

  !name of input shape file and number of media in the file
  shape_file = './shape.cube'
  media_num  = 1

  !*** MEDIA INFORMATION(START) ***************************************!
  !type of media(media ID)
  media_type(1) = 'lorentz-drude'
  !--- Au described by Lorentz-Drude model ---------------------------!
  ! The parameters are determined from:                               !
  ! (https://www.osapublishing.org/ao/abstract.cfm?uri=ao-37-22-5271) !
  !--------------------------------------------------------------------!
```

(continues on next page)

(continued from previous page)

```
  !number of poles and plasma frequency of media(media ID)
  pole_num_ld(1) = 6
  omega_p_ld(1)  = 9.030d0

  !oscillator strength, collision frequency,
  !and oscillator frequency of media(media ID,pole ID)
  f_ld(1,1:6)     = 0.760d0, 0.024d0, 0.010d0, 0.071d0, 0.601d0, 4.384d0
  gamma_ld(1,1:6) = 0.053d0, 0.241d0, 0.345d0, 0.870d0, 2.494d0, 2.214d0
  omega_ld(1,1:6) = 0.000d0, 0.415d0, 0.830d0, 2.969d0, 4.304d0, 13.32d0
  !*** MEDIA INFORMATION(END) ************************************!

  !*** SOURCE INFORMATION(START) *********************************!
  !type of method to generate the incident pulse
  !('source': incident current source)
  wave_input = 'source'

  !location of source(x,y,z)
  source_loc1(1:3) = -37.8d0, 0.0d0, 0.0d0

  !propagation direction of the incidenty pulse(x,y,z)
  ek_dir1(1:3) = 1.0d0, 0.0d0, 0.0d0
  !*** SOURCE INFORMATION(END) ***********************************!

  !*** OBSERVATION INFORMATION(START) *****************************!
  !number of observation points
  obs_num_em = 1

  !time step interval for sampling
  obs_samp_em = 20

  !location of observation point(observation ID,x,y,z)
  obs_loc_em(1,1:3) = 0.0d0, 0.0d0, 0.0d0

  !output flag for electrmagnetic field distribution(observation ID)
  yn_obs_plane_em(1) = 'n'
  !--- Make of animation file ---------------------------------------!
  ! When yn_obs_plane_em(1) = 'y', animation file can be made         !
  ! by program 'FDTD_make_figani' in SALMON utilities.                !
  ! The animation file visualizes electromagnetic field distributions !
  ! on the cross-section including the observation point              !
  ! whose location is determined by obs_loc_em.                       !
  !-------------------------------------------------------------------!
  !*** OBSERVATION END(START) ************************************!
/
```

`al_em(1:3) = 120d0, 120d0, 120d0` specifies the lengths of three sides of the rectangular parallelepiped where the grid points are prepared.

`dl_em(1:3) = 1.2d0, 1.2d0, 1.2d0` specifies the grid spacings in three Cartesian directions.

`dt_em = 2.30d-4` specifies the time step of the time evolution calculation. If you do not specifies `dt_em`, this input keyword is automatically specified by the Courant-Friedrichs-Lewy Condition.

`nt_em = 20000` specifies the number of time steps in the calculation.

`shape_file = 'shape.cube'` indicates the filename of the shape file.

`media_num = 1` specifies the number of the types of media described by the shape file('shape.cube').

`media_type(1) = 'lorentz-drude'` specifies the type of media as the Lorentz-Drude model.

`omega_p_ld(1) = 9.030d0, f_ld(1,1:6) = 0.760d0, 0.024d0, 0.010d0, 0.071d0, 0.601d0, 4.384d0, gamma_ld(1,1:6) = 0.053d0, 0.241d0, 0.345d0, 0.870d0, 2.494d0, 2.214d0,` and `omega_ld(1,1:6) = 0.000d0, 0.415d0, 0.830d0, 2.969d0, 4.304d0, 13.32d0` specify the plasma frequency, oscillator strength, collision frequency, and oscillator frequency of media, respectively.

`wave_input = 'source'` specifies an electric current source that is used for generating the pulse.

`source_loc1(1:3) = -37.8d0, 0.0d0, 0.0d0` specifies the coordinate of the current source.

`ek_dir1(1:3) = 1.0d0, 0.0d0, 0.0d0` specifies the propagation direction of the pulse (x,y,z).

`obs_num_em = 1` specifies the number of the observation point.

`obs_samp_em = 20` specifies the sampling number for time steps. In this case, output files are generated every 20 steps.

`obs_loc_em(1,1:3) = 0.0d0, 0.0d0, 0.0d0` specifies the coordinate of the current source.

`yn_obs_plane_em(1) = 'n'` determines to output the electrmagnetic fields on the planes (xy, yz, and xz planes) for the observation point. This option must be `'y'` for generating animation files by using SALMON utilities: https://salmon-tddft.jp/utilities.html

See &maxwell in *List of all input keywords* for more information.

Output files

After the calculation, following output files are created in the directory `'result'`,

file name	description
obs0_info.data	information to generate animation
obs1_at_point_rt.data	components of electric and magnetic fields as functions of time

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_0/exercise_zip_files/10_classicEM_rt.zip

Explanations of the files are described below:

obs0_info.data

This file is used to generate animation files by using SALMON utilities: https://salmon-tddft.jp/utilities.html

obs1_at_point_rt.data

Results of time evolution calculation for electric and magnetic fields at observation point 1.

```
# Real time calculation:
# E: Electric field
# H: Magnetic field
# 1:Time[fs] 2:E_x[V/Angstrom] 3:E_y[V/Angstrom] 4:E_z[V/Angstrom] 5:H_x[A/Angstrom]␣
↪6:H_y[A/Angstrom] 7:H_z[A/Angstrom]
```

CHAPTER

FOUR

INPUT KEYWORDS FOR EXERCISES

We here summarize input keywords that appear in *Exercise*. A thorough list of the input keywords can be found in the downloaded file in *List of all input keywords*.

4.1 &calculation

Mandatory: theory

```
&calculation
  theory = 'dft'
/
```

The value of the `theory` should be one of `'dft'`, `'dft_md'`, `'tddft_response'`, `'tddft_pulse'`, `'single_scale_maxwell_tddft'`, `'multi_scale_maxwell_tddft'`, `'maxwell'`, and `'dft_k_expand'`.

`dft` : ground state calculation based on DFT

`dft_md` : adiabatic ab initio MD simulations based on DFT

`tddft_response` : simulations based on TDDFT to obtain linear-response

`tddft_pulse` : simulations based on TDDFT using pulsed light

`single_scale_maxwell_tddft` : coupled Maxwell and TDDFT single-scale simulation

`multi_scale_maxwell_tddft` : coupled Maxwell and TDDFT multi-scale simulation

`maxwell` : electromagnetic simulations based on the Maxwell's equations

`dft_k_expand` : convert checkpoint data of dft with k-points calculation to that of larger supercell system with gamma-point

4.2 &units

Mandatory: none

```
&units
  unit_system = 'A_eV_fs'
/
```

This namelist specifies the unit system to be used in the input file. Options are 'A_eV_fs' for Angstrom, eV, and fs, and 'a.u.' or 'au' for atomic units. If you do not specify it, atomic unit will be used as default.

For isolated systems (specified by `yn_periodic = 'n'` in `&system`), the unit of 1/eV is used for the output files of DOS and PDOS if `unit_system = 'A_eV_fs'` is specified, while atomic unit is used if not. For other output files, the Angstrom/eV/fs units are used irrespective of the input keyword.

For periodic systems (specified by `yn_periodic = 'y'` in `&system`), the unit system specified by this input keyword is used for most output files. See the first few lines of output files to confirm the unit system adopted in the file.

4.3 &control

Mandatory: none

```
&control
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files. If you do not specify it, the file name will start with 'default'.

4.4 &system

Mandatory: yn_periodic, al, nelem, natom, nelec, nstate

For an isolated molecule (Exercises-1, 2, 3, 8, 9):

```
&system
  yn_periodic = 'n'
  al(1:3) = 16.0d0, 16.0d0, 16.0d0
  nelem = 2
  natom = 4
  nelec = 10
  nstate = 6
/
```

`yn_periodic = 'n'` indicates that the isolated boundary condition will be used in the calculation. `al(1:3) = 16.0d0, 16.0d0, 16.0d0` specifies the lengths of three sides of the rectangular parallelepiped where the grid points are prepared. `nelem = 2` and `natom = 4` indicate the number of elements and the number of atoms in the system, respectively. `nelec = 10` indicate the number of valence electrons in the system.

`nstate = 6` indicates the number of Kohn-Sham orbitals to be solved.

`nstate` should be equal to or larger than `nelec/2`.

For a periodic system (Exercises-4, 5, 6, 7):

```
&system
  yn_periodic = 'y'
  al(1:3) = 10.26d0, 10.26d0, 10.26d0
  nelem = 1
  natom = 8
  nelec = 32
  nstate = 32
/
```

`yn_periodic = 'y'` indicates that three dimensional periodic boundary condition (bulk crystal) is assumed. `al(1:3) = 10.26d0, 10.26d0, 10.26d0` specifies the lattice constans of the unit cell. `nelem = 1` and `natom = 8` indicate the number of elements and the number of atoms in the system, respectively. `nelec = 32` indicate the number of valence electrons in the system. `nstate = 32` indicates the number of Kohn-Sham orbitals to be solved.

4.5 &pseudo

Mandatory: pseudo_file, izatom

For C2H2 molecule:

```
&pseudo
  file_pseudo(1) = './C_rps.dat'
  file_pseudo(2) = './H_rps.dat'
  izatom(1) = 6
  izatom(2) = 1
  lloc_ps(1) = 1
  lloc_ps(2) = 0
/
```

Parameters related to atomic species and pseudopotentials. `pseudo_file(1) = './C_rps.dat'` indicates the filename of the pseudopotential of element. `izatom(1) = 6` specifies the atomic number of the element.

For crystalline Si:

```
&pseudo
  file_pseudo(1) = './Si_rps.dat'
  izatom(1) = 14
  lloc_ps(1) = 2
/
```

`file_pseudo(1) = './Si_rps.dat'` indicates the pseudopotential filename of element. `izatom(1) = 14` indicates the atomic number of the element.

4.6 &functional

```
&functional
  xc ='PZ'
/
```

`xc ='PZ'` indicates that (adiabatic) local density approximation is adopted (Perdew-Zunger: Phys. Rev. B23, 5048 (1981)). This is the default choice.

For isolated systems (specified by `yn_periodic = 'n'` in `&system`), only the default choice of 'PZ' is available at present.

For periodic systems (specified by ``yn_periodic = 'y'` in `&system`), the following functionals may be available in addition to 'PZ', `xc = 'PZM'`

Perdew-Zunger LDA with modification to improve sooth connection between high density form and low density one, `xc = 'TBmBJ' cval = 1.0` :J. P. Perdew and Alex Zunger, Phys. Rev. B 23, 5048 (1981).

Tran-Blaha meta-GGA exchange with Perdew-Wang correlation. :Fabien Tran and Peter Blaha, Phys. Rev. Lett. 102, 226401 (2009). John P. Perdew and Yue Wang, Phys. Rev. B 45, 13244 (1992). This potential is known to provide

a reasonable description for the bandage of various insulators. For this choice, the additional mixing parameter 'cval' may be specified. If cval is set to a minus value, the mixing-parameter will be computed following the formula in the original paper [Phys. Rev. Lett. 102, 226401 (2009)]. The default value for this parameter is 1.0.

Since version 1.1.0, exchange-correlation functionals in Libxc library (http://www.tddft.org/programs/libxc/) have been usable in SALMON. At present, usable functionals are limited to LDA and GGA. For periodic systems, meta-GGA functionals are usable as well. To specify the exchange-correlation potentials of Libxc library, there are two ways. If the exchange and correlation potentials are given separately, you need to specify both `alibx` and `alibc` separately. If the exchange and correlation potentials are given as a combined set, you need to specify `alibxc`. We show below an example:

```
&functional
  alibx = 'LDA_X'
  alibc = 'LDA_C_PZ'
/
```

Available sets of the functionals are listed at the website http://www.tddft.org/programs/libxc/functionals/ .

Note that, the hybrid functionals (hybrid gga/mgga) are not supported in the current (version 2.0.0) of SALMON.

4.7 &rgrid

Mandatory: dl or num_rgrid

`dl(1:3) = 0.25d0, 0.25d0, 0.25d0` specify the grid spacing in three Cartesian coordinates. This is adopted for C2H2 calculation (Excercises-1, 2, 3, 8, 9).

```
&rgrid
  dl(1:3) = 0.25d0, 0.25d0, 0.25d0
/
```

`num_rgrid(1:3) = 12, 12, 12` specify the number of grid points in each Cartesian direction. This is adopted for crystalline Is calculation (Excercises-4, 5, 6, 7).

```
&rgrid
  num_rgrid(1:3) = 12, 12, 12
/
```

4.8 &kgrid

Mandatory: none

This group provides grid spacing of k-space for periodic systems.

```
&kgrid
  num_kgrid(1:3) = 4, 4, 4
/
```

4.9 &scf

Mandatory: nscf

This group specifies parameters related to the self-consistent field calculation.

```
&scf
  nscf = 200
  threshold = 1.0d-9
/
```

`nscf = 200` is the number of scf iterations in the ground state calculation. the scf loop in the ground state calculation ends before the number of the scf iterations reaches `nscf`, if a convergence criterion is satisfied.

4.10 &analysis

Mandatory: none

The following input keywords specify whether the output files are created or not after the calculation. In the ground state calculation of isolated systems (Excercise-1):

```
&analysis
  yn_out_psi  = 'y'
  yn_out_dns  = 'y'
  yn_out_dos  = 'y'
  yn_out_pdos = 'y'
  yn_out_elf  = 'y'
/
```

In the following input keywords, variables related to time-frequency Fourier analysis are specified.

```
&analysis
  de = 1.0d-2
  nenergy = 3000
/
```

`de = 1.0d-2` specifies the energy spacing, and `nenergy = 3000` specifies the number of energy steps in the time-frequency Fourier transformation.

4.11 &tgrid

Mandatory: dt, nt

```
&tgrid
  dt = 1.25d-3
  nt = 5000
/
```

`dt = 1.25d-3` specifies the time step of the time evolution calculation. `nt = 5000` specifies the number of time steps in the calculation.

4.12 &emfield

This group specifies the pulse shape of an electric filed applied to the system in time evolution calculations. We explain below separating two cases, *Linear response calculations* and *Pulsed electric field calculations*.

4.12.1 Linear response calculations

A weak impulsive field is applied at $t = 0$. For this case, `ae_shape1 = 'impulse'` should be described.

Mandatory: ae_shape1

```
&emfield
  ae_shape1 = 'impulse'
  epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
/
```

`epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0` specifies a unit vector that indicates the direction of the impulse.

For a periodic system specified by `yn_periodic = 'y'`, one may add `trans_longi`. It has the value, `'tr'`(transverse) or `'lo'`(longitudinal), that specifies the treatment of the polarization in the time evolution calculation. The default is `'tr'`.

The magnitude of the impulse of the pulse may be explicitly specified by, for example, `e_impulse = 1.00d-2`. The default is '1.00d-2' in atomic unit.

4.12.2 Pulsed electric field calculations

A Pulsed electric field of finite time duration is applied. For this case, `as_shape1` should be specified. It indicates the shape of the envelope of the pulse. The options include 'Acos2' and 'Ecos2' (See below for other options).

Mandatory: ae_shape1, {I_wcm2_1 or E_amplitude1}, tw1, omega1, epdir_re1, phi_cep1

```
&emfield
  ae_shape1 = 'Ecos2'
  I_wcm2_1 = 1.00d8
  tw1 = 6.00d0
  omega1 = 9.28d0
  epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0
  phi_cep1 = 0.75d0
/
```

`ae_shape1 = 'Ecos2'` specifies the envelope of the pulsed electric field, 'Ecos2' for the cos^2 envelope for the electric field. If 'Acos2' is specified, this gives cos^2 envelope for the vector potential. Note that 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish. There is no such restriction for the 'Acos2' pulse shape.

`I_wcm2_1 = 1.00d8` specifies the maximum intensity of the applied electric field in unit of W/cm^2. It is also possible to specify the maximum intensity of the pulse by `E_amplitude1`.

`tw1 = 6.00d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`omega1 = 9.28d0` specifies the average photon energy (frequency multiplied with hbar).

`epdir_re1(1:3) = 0.00d0, 0.00d0, 1.00d0` specifies the real part of the unit polarization vector of the pulsed electric field. If only the real part is specified, it describes a linearly polarized pulse. Using both real ('epdir_re1') and imaginary ('epdir_im1') parts of the polarization vector, circularly (and general ellipsoidary) polarized pulses may be described.

`phi_cep1 = 0.75d0` specifies the carrier envelope phase of the pulse. As noted above, 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish. There is no such restriction for the 'Acos2' pulse shape.

It is possible to use two pulses simultaneously to simulate pump-probe experiments, adding information for two pulses. To specify the second pulse, change from 1 to 2 in the input keywords, like `ae_shape2`. The time delay between two pulses is specified by the input keyword 't1_t2'.

For a periodic system specified by `yn_periodic = 'y'`, one may add `trans_longi`. It has the value, `'tr'`(transverse) or `'lo'`(longitudinal), that specifies the treatment of the polarization in the time evolution calculation. The default is `'tr'`. For a periodic system, it is also specify 'Acos3', 'Acos4', 'Acos6', 'Acos8' for `ae_shape1`.

4.13 &multiscale

This group specifies information necessary for Maxwell-TDDFT multiscale calculations.

```
&multiscale
  nx_m = 8
  ny_m = 1
  nz_m = 1
  hx_m = 100.0d0
  hy_m = 100.0d0
  hz_m = 100.0d0
  nxvacl_m = 1000
  nxvacr_m = 1000
/
```

`nx_m = 8` specifies the number of the macroscopic grid points for x-direction in the spatial region where the material exists. `ny_m = 1` and `nz_m = 1` are those for y- and z-directions.

`hx_m = 100.0d0` specifies the grid spacing of the macroscopic grid for x-direction. `hy_m = 100.0d0` and `hz_m = 100.0d0` are those for y- and z-directions.

`nxvacl_m = 1000` and `nxvacr_m = 1000` indicate the number of grid points in the vacuum region, `nxvacl_m` for the left and `nxvacr_m` for the right from the surface of the material.

4.14 &atomic_coor

Mandatory: atomic_coor or atomic_red_coor

For C2H2 molecule:

```
&atomic_coor
  'C'    0.000000    0.000000    0.599672  1
  'H'    0.000000    0.000000    1.662257  2
  'C'    0.000000    0.000000   -0.599672  1
  'H'    0.000000    0.000000   -1.662257  2
/
```

Cartesian coordinates of atoms. The first column indicates the element. Next three columns specify Cartesian coordinates of the atoms. The number in the last column labels the element.

4.15 &atomic_red_coor

Mandatory: atomic_coor or atomic_red_coor

For a crystalline silicon:

```
&atomic_red_coor
  'Si'        .0      .0      .0      1
  'Si'        .25     .25     .25     1
  'Si'        .5      .0      .5      1
  'Si'        .0      .5      .5      1
  'Si'        .5      .5      .0      1
  'Si'        .75     .25     .75     1
  'Si'        .25     .75     .75     1
  'Si'        .75     .75     .25     1
/
```

Cartesian coordinates of atoms are specified in a reduced coordinate system. First column indicates the element, next three columns specify reduced Cartesian coordinates of the atoms, and the last column labels the element.

CHAPTER

FIVE

HOW TO CITE SALMON

5.1 Suggested Citations

If you publish a paper in which SALMON makes an important contribution, please cite the SALMON code paper, Ref. [1] published in Computer Physics Communications.

We also suggest you to cite the following papers depending on your usage of SALMON.

- If you use SALMON for electron dynamics calculations of a large-size system, Ref. [2] that discusses massively parallel implementation utilizing spatial divisions will be appropriate.
- if you use SALMON to calculate electron dynamics in a unit cell of crystalline solid, Ref. [3] discussing formalism and numerical implementation will be appropriate.
- Ref. [4] is one of the first implementations of the real-time time-dependent density functional calculation, in particular, instantaneous kick for the linear response calculations.
- If you use multiscale calculation coupling Maxwell equations for the electromagnetic fields of light and electron dynamics, Ref. [5] discussing the formalism and the numerical implementation will be appropriate.
- Ref. [6] describes parallelization method for the coupled Maxwell - TDDFT calculations.
- Ref. [7] describes computational aspects of electron dynamics calculations for periodic systems in many-core processors:

Appendix:

CHAPTER

SIX

LIST OF ALL INPUT KEYWORDS

6.1 &calculation

- **theory (character, default='none')**

  Choice of Calculation theories.

Options

`dft` / ground state calculation based on DFT

`dft_md` / adiabatic ab initio MD simulations based on DFT

`tddft_response` / simulations based on TDDFT for response

`tddft_pulse` / simulations based on TDDFT using pulsed light

`single_scale_maxwell_tddft` / coupled Maxwell and TDDFT single-scale simulation

`multi_scale_maxwell_tddft` / coupled Maxwell and TDDFT multi-scale simulation

`maxwell` / electromagnetic simulations based on the Maxwell's equations

`dft_k_expand` / convert checkpoint data of dft with k-points calculation to that of larger supercell system with gamma-point

- **yn_md (character, Default='n')[Trial]**

  Available for `theory='dft'` (Adiabatic ground-state MD) and `theory='tddft_pulse'` (Ehrenfest MD).

  Molecular dynamics option.

  Options

  `'y'` / enable

  `'n'` / disable

- **yn_opt (character, Default='n')[Trial]**

  Available for `theory='dft'`.

  Geometry optimization option.

  Options

  `'y'` / enable

  `'n'` / disable

6.2 &control

- **sysname (character, Default='default')**

  Available for all options of `theory`.

  Name of calculation. This is used for a prefix of output files.

- **base_directory (character, Default='./')**

  Available for all options of `theory`.

  Name of a default directory, where the basic results will be written down.

- **yn_restart (character, Default='n')**

  Available for `theory='dft*' or '*tddft*'`.

  Restart option.

  Enable(`'y'`)/disable(`'n'`).

- **directory_read_data (character, Default='restart/')**

  Directory name for the restart data that is written down in the previous run

- **yn_self_checkpoint (character, Default='n')**

  If set *'y'*: When saving intermediate results of the simulation (this call checkpointing), each process write/read a checkpoint data independently.

  This option helps large-scale simulation to recover from system failure, which reduce restart costs.

- **checkpoint_interval (integer, Default=-1)**

  Available for `theory='dft*' or '*tddft*'`.

  Interval of time step (or iteration step) of writing down check-point data during the time-propagation or iteration. These are not written down If negative value is set.

- **yn_reset_step_restart (character, Default='n')**

  Available for `yn_restart='y'` with the DFT/TDDFT based options of `theory`.

  In the case of restarting, the initial step of SCF iteration (for DFT) or time step (for TDDFT) are reset to 0 at begining. Then, the memory of the density in the previous SCF iteration steps (in GS) is abondoned.

- **read_gs_restart_data (character, Default='all')**

  Available for `yn_restart='y'` with `theory='dft'`.
  Options
  `all` / all of restart data are read
  `rho_inout` / only electron densities including memories at previous iteration steps are read
  `rho` / only the latest electron density is read (user-made data)
  `wfn` / only wavefunctions is read

  Specified data which is included in the restart (or checkpoint) directory generated in the previous calculation is used for restarting SCF iteration in DFT. The default option `'all'` gives the complete restart. The other options use a part of restart data (other necessary data is generated as done in the initial SCF step)

- **write_gs_restart_data (character, Default='all')**

  Available for `theory='dft'`.
  Options
  `all` / all of restart data are written out
  `rho_inout` / only electron densities including memories at previous iteration steps are written out
  `wfn` / only wavefunctions is written out

  Specified data is written out in the restart (or checkpoint) directory. The default option `'all'` gives the complete set of restart data.

- **time_shutdown (real(8), Default=-1d0)[Trial]**

  Available for `theory='dft' or '*tddft*'`.

  Timer for automatic shutdown. The unit is second. If negative time is chosen, the automatic shutdown is not performed.

- **method_wf_distributor (character, Default='single')**

  Available for `theory='dft*' or '*tddft*'`.
  Select a method of save/load the wave function.
  'single': wave function saves/loads a single shared file.
  'slice' : wave function saves/loads to a file per the orbital function.
  'slice' reduces I/O costs, and they can helps flexible large-scale simulation.

- **nblock_wf_distribute (integer, Default='16')**

  Available for `method_wf_distributor='slice'`.
  'slice' mode saves `nblock_wf_distribute`-files to a directory.
  In a default, they will saves 16 files to same directory.

6.3 &units

- **unit_system (character, Default='au')**

  Units of input variables.
  Options
  `'au'` or `'a.u.'` / atomic unit system.
  `'A_eV_fs'` / Angstrom-eV-fs unit system

6.4 &parallel

- **nproc_k/nproc_ob/nproc_rgrid(3) (integer, Default=0)**

  Old infomation: 0d
  Options
  `nproc_k`/ Number of MPI parallelization for orbitals that related to the wavefunction calculation.
  `nproc_ob`/ Number of MPI parallelization for orbitals that related to the wavefunction calculation.
  `nproc_rgrid(3)'`/ Number of MPI parallelization for each direction in real-space that related to the wavefunction and the electron density calculations.

  Defaults are `0` for `nproc_k`/`nproc_ob` and `(0,0,0)` for `nproc_rgrid`. If users use the defaults, automatic proccess assignment is done. Users can also specify `nproc_k`, `nproc_ob`, and `nproc_rgrid` manually. In that case, `nproc_k` must be set to `1` for isolated system calculations. `nproc_k` and `nproc_k` must be set to `1` for `theory='maxwell'`. In addition, followings must be satisfied.

  `nproc_k * nproc_ob * nproc_rgrid(1) * nproc_rgrid(2) * nproc_rgrid(3) =` total number of processes.

- **yn_ffte (character, Default='n')**

  Available for `&system/yn_periodic='y'`
  Old infomation: 0d
  Method of Fourier transformation.
  Enable(`'y'`)/disable(`'n'`).
  SALMON uses FFT (via FFTE library) to solve poisson equation.
  When enabling it, followings must be satisfied.

  ```
  mod(num_rgrid(1), nproc_rgrid(2)) == 0
  mod(num_rgrid(2), nproc_rgrid(2)) == 0
  mod(num_rgrid(2), nproc_rgrid(3)) == 0
  mod(num_rgrid(3), nproc_rgrid(3)) == 0
  ```

- **yn_scalapack (character, Default='n')**

  Available for `&calculation/theory='dft' or 'dft_md'`
  SALMON uses ScaLAPACK library to solve eigenvalue problem in subspace diagonalization.
  When enabling it, you should build SALMON by linking ScaLAPACK library.

- **yn_eigenexa (character, Default='n')**

Available for `&calculation/theory='dft' or 'dft_md'`
SALMON uses RIKEN R-CCS EigenExa library to solve eigenvalue problem in subspace diagonalization.
When enabling it, you should build SALMON by linking ScaLAPACK and EigenExa libraries.

- **yn_diagonalization_red_mem (character, Default='n')**

  Available for `&parallel/yn_scalapack='y'` or `&parallel/yn_eigenexa='y'`
  We use ScaLAPACK/EigenExa libraries by optimized algorithm to reduce memory consumption.

- **process_allocation (character, Default='grid_sequential')**

  Old infomation: 0d
  You can select the process allocation ordering.
  `'grid_sequential'` / real-space grid major ordering.
  `'orbital_sequential'` / orbital-space major ordering.

  Suggestion:
  `&calculation/theory='dft' or 'dft_md'` / *orbital_sequential*
  `&calculation/theory='tddft*' or '*maxwell_tddft'` / *grid_sequential*

6.5 &system

- **yn_periodic (character, Default='n')**

  Available for all options of `theory`.
  Option of periodic boundary condition.
  `'y'` / periodic systems (solids)
  `'n'` / isolated systems

- **spin (character, Default='unpolarized')**

  Available for all options of `theory` except for `theory='maxwell'`.
  Variable for classification of spin-unpolarized (closed shell) systems and spin-polarized (open shell) systems.
  Options
  `'unpolarized'` / spin-unpolarized systems (default)
  `'polarized'` / spin-polarized systems

- **al(3) (real(8), Default=0d0)**

  Available for all options of `theory` except for `theory='maxwell'`.

  Spatial grid box size or lattice constants for cuboid cell (x, y, z). For nonorthogonal cell, see `al_vec1, al_vec2, al_vec3`.

- **al_vec1(3)/al_vec2(3)/al_vec3(3) (real(8), Default=0d0)**

  Available for all options of `theory` except for `theory='maxwell'`.

  Primitive lattice vectors for nonorthogonal cell.

- **nstate (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of orbitals/bands.

- **nstate_spin(2) (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of orbitals/bands for up/down-spin electrons can be specified for each by `nstate_spin(1)/nstate_spin(2)`. This option is incompatible with `nstate` (?? does it mean nstate specified is ignored if this option is specified ??)

- **nelec (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of valence electrons.

- **nelec_spin(2) (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of up/down-spin electrons can be specified for each by `nelec_spin(1)/nelec_spin(2)`. This option is incompatible with `nelec` (?? does it mean nelec specified is ignored if this option is specified ??)

- **temperature (real(8), Default=-1d0)**

  Available for DFT-based options of `theory`
  Temperature of electrons. The value must be given by the unit of energy as specified in `&units/unit_system`.
  The kelvin unit can be used by the keyword `temperature_k` (see next).
  `temperature < 0` / the occupation numbers are fixed by `nelec` (for bandgap system).
  `temperature = 0` / redistribution of the occupation numbers by the step function.
  `temperature > 0` / redistribution of the occupation numbers by the Fermi-Dirac distribution function.

- **temperature_k (real(8), Default=-1d0)[Trial]**

  Available for DFT-based options of `theory`

  The same as `temperature` but in kelvin.

- **nelem (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of used atomic elements in the system.

- **natom (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Number of atoms in the system.

- **file_atom_red_coor (character, Default='none')[Trial]**

  Available for the DFT/TDDFT based options of `theory`.

  File name for atomic positions given in reduced coordinates. This option is incompatible with `&system/file_atom_coor`, `&atomic_coor`, and `&atomic_red_coor`.

- **file_atom_coor (character, Default='none')[Trial]**

  Available for the DFT/TDDFT based options of `theory`.

  File name for atomic Cartesian coordinates (The unit is specified by `&units/unit_system`). This option is incompatible with `&system/file_atom_coor`, `&atomic_coor`, and `&atomic_red_coor`. (XXX why this keyword is not in &atomic_coor ?? XXX)

6.6 &atomic_red_coor

Atomic coordinates in reduced coordinates as following format:

'Si' 0.00 0.00 0.00 1
'Si' 0.25 0.25 0.25 1
...

Here, the information of atoms is ordered in row. For example, the first row is for the first atom. The number of rows must be equal to `&system/natom`. The first coloum can be any caracters and does not affect calculations. The second, third and fourth columns are reduced coordinates for the first, second and third directions, respectively. The fifth column is a serial number of the atom spieces, which is defined in `&pseudo`. This option is incompatible with `&system/file_atom_red_coor`, `&system/file_atom_coor`, and `&atomic_coor`.

6.7 &atomic_coor

Cartesian atomic coordinates. The format is the same as &atomic_red_coor. The unit can be chosen by `&units/unit_length`. This option is incompatible with `&system/file_atom_red_coor`, `&system/file_atom_coor`, and `&atomic_red_coor`.

6.8 &pseudo

Input for psudopotentials. Size of array (:) is equal to `&system/nelem`.

- **izatom(:) (integer, Default=-1)**

  Available for the DFT/TDDFT based options of `theory`.

  Atomic number.

- **file_pseudo(:) (character, Default='none')**

  Available for the DFT/TDDFT based options of `theory`.

  File name for pseudopotential.

- **lmax_ps(:) (integer, Default=-1)**

  Available for the DFT/TDDFT based options of `theory`.

  Maximum angular momentum of pseudopotential projectors. If not given, it is automatically read from the pseudopotential file.

- **lloc_ps(:) (integer, Default=-1)**

  Available for the DFT/TDDFT based options of `theory`.

  Angular momentum of pseudopotential that will be treated as local.

- **yn_psmask(:) (character, Default='n')[Trial]**

  Available for the DFT/TDDFT based options of `theory`.
  Fourier filtering for pseudopotentials.

  Enable(`'y'`)/disable(`'n'`)

- **alpha_mask(:) (real(8), Default=0.8d0)[Trial]**

  Available for the DFT/TDDFT based options of `theory`.

  Parameter for the Fourier filtering for pseudopotential.

- **gamma_mask(:) (real(8), Default=1.8d0)[Trial]**

  Available for the DFT/TDDFT based options of `theory`.

  Parameter for the Fourier filtering for pseudopotential.

- **eta_mask(:) (real(8), Default=15.0d0)[Trial]**

  Available for the DFT/TDDFT based options of `theory`.

  Parameter for the Fourier filtering for pseudopotential.

6.9 &functional

- **xc (character, Default='none')**

  Available for the DFT/TDDFT based options of `theory`.
  Exchange-correlation functionals.
  At present version, the functional 'PZ', 'PZM' and 'TBmBJ' is available for both 0d/3d calculations, and the functionals 'TPSS' and 'VS98' are available for 3d calculations. (XXX need check XXX)
  Options

  `'PZ'`: Perdew-Zunger LDA :Phys. Rev. B 23, 5048 (1981).

  `'PZM'`: Perdew-Zunger LDA with modification to improve sooth connection between high density form and low density one. :J. P. Perdew and Alex Zunger, Phys. Rev. B 23, 5048 (1981).

  `'TBmBJ'`: Tran-Blaha meta-GGA exchange with Perdew-Wang correlation. :Fabien Tran and Peter Blaha, Phys. Rev. Lett. 102, 226401 (2008). John P. Perdew and Yue Wang, Phys. Rev. B 45, 13244 (1992).

  `'TPSS'`: Tao, Perdew, Staroverov and Scuseria meta-GGA exchange correlation. :J. Tao, J. P. Perdew, V. N. Staroverov, and G. E. Scuseria, Phys. Rev. Lett. 91, 146401 (2003).

  `'VS98'`: van Voorhis and Scuseria exchange with Perdew-Wang correlation: T. Van Voorhis and G. E. Scuseria, J. Chem. Phys. 109, 400 (1998).

- **cname, xname (character, Default='none')**

  Available for `theory='XXX'`.
  XXX

- **alibxc, alibx, alibc (character, Default='none')**

  Available for the DFT/TDDFT based options of `theory`.

  By specifying `alibxc`, the functionals prepared in libxc package are available. They can be set indivisually by specifying `alibx` and `alibc`. To use libxc libraries, `--with-libxc` option must be added in excecuting configure. The available option of the exchange-correlation functionals are listed in the LibXC website. [See http://www.tddft.org/programs/libxc/functionals/]

- **cval (real(8), Default=-1d0)**

  Available for `xc='TBmBJ'`.

  Mixing parameter in Tran-Blaha meta-GGA exchange potential. If `cval` is set to a minus value, the mixing-parameter computed by the formula in the original paper [Phys. Rev. Lett. 102, 226401 (2008)]. Default is estimated from $\langle |\nabla \rho(\mathbf{r};t)| / \rho(\mathbf{r};t) \rangle$.

6.10 &rgrid

- **dl(3) (real(8), Default=0d0)**

  Available for the DFT/TDDFT based options of `theory`.

  Spacing of real-space grids. (This cannot be used together with `&rgrid/num_rgrid`.)

- **num_rgrid(3) (integer, Default=0)**

  Available for the DFT/TDDFT based options of `theory`.

  Dividing number of real-space grids for each direction. (This cannot be used together with `&rgrid/dl`.)

6.11 &kgrid

- **num_kgrid(3) (integer, Default=1)**

  Available for `yn_periodic='y'` with the DFT/TDDFT based options of `theory`.
  Number of k-points (grid points of k-vector) for each direction discretizing the Brillouin zone.

- **file_kw (character, Default='none')**

  Available for `yn_periodic='y'` with the DFT/TDDFT based options of `theory`.
  File name for user specified k-points. This file will be read if `num_kgrid` is smaller than 1. The k-points are given as following format, for example, :

  ```
  8 #(number of k-points)
  1 -0.50 -0.50 -0.50 0.1250 #(id, kx, ky, kz, weight)
  2 -0.50 -0.50 0.00 0.1250
  3 -0.50 0.00 -0.50 0.1250
  4 -0.50 0.00 0.00 0.1250
  5 0.00 -0.50 -0.50 0.1250
  6 0.00 -0.50 0.00 0.1250
  7 0.00 0.00 -0.50 0.1250
  8 0.00 0.00 0.00 0.1250
  ```

6.12 &tgrid

- **nt (integer, Default=0)**

  Available for 'dft_md' and TDDFT-based options of `theory`.

  Number of total time steps for real-time propagation.

- **dt (real(8), Default=0d0)**

  Available for 'dft_md' and TDDFT-based options of `theory`.

  Time step size.

- **gram_schmidt_interval (integer, Default=-1)**

  Available for TDDFT-based options of `theory`.

Interval of time step for the Gram-Schmidt orthonormalization of the orbital wavefunctions in the time-evolution calculation. If this is set to zero, it is used at the initial step only.

6.13 &propagation

- **n_hamil (integer, Default=4)**

  Available for TDDFT-based options of `theory`.
  Order of Taylor expansion of a propagation operator.

- **propagator (character, Default=middlepoint')**

  Available for TDDFT-based options of `theory`.
  Propagator (time-integrator).
  Options
  `middlepoint` / propagator with the Hamiltoinan at midpoint of two-times.
  `aetrs` / time-reversal symmetry propagator.
  [M.A.L. Marques, A. Castro, G.F. Bertsch, and A. Rubio, Comput. Phys. Commun., 151 60 (2003)].

- **yn_predictor_corrector (character(1), Default='n')**

  Available for TDDFT-based options of `theory`.
  Switch of the predictor-corrector method of TDDFT.
  For meta-GGA functionals (`xc='tbmbj'` or `'bj_pw'`), the predictor corrector is automatically used even with `yn_predictor_corrector='n'`.
  Options
  `'y'` / enable
  `'n'` / disable

- **yn_fix_func (character(1), Default='n')[currently not available]**

  Available for 'dft_md' and TDDFT-based options of `theory`.
  Option not to update functional (or Hamiltonian) in time-evolution, i.e., keep ground state Hamiltonian. (currently not available)
  Options
  `'y'` / enable
  `'n'` / disable

6.14 &scf

- **method_init_wf (character, Default='gauss')**

  Available for 'dft' and 'dft_md' options of `theory`.
  The generation method of the initial wavefunction (orbital) at the begening of the SCF iteration in DFT calculation.
  Options
  `gauss` / put single gauss function using a random number on each initial orbital
  `gauss2` / put two gauss functions using a random number on each initial orbital
  `gauss3` / put three gauss functions using a random number on each initial orbital
  `gauss4` / put four gauss functions using a random number on each initial orbital
  `gauss5` / put five gauss functions using a random number on each initial orbital

`gauss10` / put ten gauss functions using a random number on each initial orbital
`random` / give a random number at each real-space grid point on each initial orbital

- **iseed_number_change (integer, Default=0)**

  Available for 'dft' and 'dft_md' options of `theory`.

  The seed of the random numbers are changed by adding the given number for generating the initial wavefunctions.

- **nscf (integer, Default=300)**

  Available for 'dft' and 'dft_md' options of `theory`.

  Number of maximum SCF cycle in DFT calculation.

- **method_min (character, Default='cg')**

  Available for 'dft' and 'dft_md' options of `theory`.
  Method for SCF iteration
  Options
  `cg` / Conjugate-Gradient(CG) method
  `diis` / DIIS method
  `cg-diis` / CG-DIIS method

- **ncg (integer, Default=4)**

  Available for 'dft' and 'dft_md' options of `theory`.

  Number of interation of Conjugate-Gradient method for each scf-cycle.

- **ncg_init (integer, Default=4)**

  Available for 'dft' and 'dft_md' options of `theory`.

  Number of interation of Conjugate-Gradient method for the first SCF step.

- **method_mixing (character, Default='broyden')**

  Available for 'dft' and 'dft_md' options of `theory`.
  Methods for density/potential mixing for scf cycle.
  Options
  `simple` / Simple mixing method
  `broyden` / modified-Broyden method
  `pulay` / Pulay method

- **mixrate (real(8), Default=0.5d0)**

  Available for `method_mixing='simple'` in 'dft' and 'dft_md' options of `theory`.
  Mixing ratio for simple mixing.

- **nmemory_mb (integer, Default=8)**

  Available for `method_mixing='broyden'` in 'dft' and 'dft_md' options of `theory`.
  Number of previous densities to be stored in SCF iteration cycle for the modified-Broyden method. This must be less than 21.

- **alpha_mb (real(8), Default=0.75d0)**

  Available for `method_mixing='broyden'` in 'dft' and 'dft_md' options of `theory`.
  Parameter of the modified-Broyden method.

- **nmemory_p (integer, Default=4)**

Available for `method_mixing='pulay'` in 'dft' and 'dft_md' options of `theory`.
Number of previous densities to be stored in SCF iteration cycle for the Pulay method.

- **beta_p (real(8), Default=0.75d0)**

  Available for `method_mixing='pulay'` in 'dft' and 'dft_md' options of `theory`.
  Parameter of the mixing rate for the Pulay method.

- **yn_auto_mixing (character, Default='n')**

  Available for 'dft' and 'dft_md' options of `theory`.
  The option to change the mixing-rate automatically (i.e. automatic adjustments of `mixrate`/`alpha_mb`/`beta_p`)
  Options
  `'y'` / enable
  `'n'` / disable

- **update_mixing_ratio (real(8), Default=3.0d0)**

  Available for `yn_auto_mixing='y'` in 'dft' and 'dft_md' options of `theory`.
  Threshold for the change of the mixing-rate in `yn_auto_mixing='y'` option. The mixing-rate is reduced to half when the ratio of the density differences between the current and previous iteration steps is larger than `update_mixing_ratio`.

- **yn_subspace_diagonalization (character, Default='y')**

  Available for 'dft' and 'dft_md' options of `theory`.
  Option of subspace diagonalization during SCF cycle.
  Options
  `'y'` / enable
  `'n'` / disable

- **convergence (character, Default='rho_dne')**

  Available for 'dft' and 'dft_md' options of `theory`.
  Quantity that is used for convergence check in SCF calculation.
  Options
  `'rho_dne'`/ Convergence is checked by sum_ix|rho(ix,iter)-rho(ix,iter-1)|dx/N, where iter is iteration number of SCF calculation and N is `&system/nelec`, the number of the valence electrons.
  `'norm_rho'`/ Convergence is checked by the square of the norm of difference of density, $||$rho_iter(ix)-rho_iter-1(ix)$||^2$=sum_ix|rho(ix,iter)-rho(ix,iter-1)|^2.
  `'norm_rho_dng'`/ Convergence is checked by $||$rho_iter(ix)-rho_iter-1(ix)$||^2$/(number of grids). "dng" means "devided by number of grids".
  `'norm_pot'`/ Convergence is checked by $||$Vlocal_iter(ix)-Vlocal_iter-1(ix)$||^2$, where Vlocal is Vh + Vxc + Vps_local.
  `'pot_dng'`/ Convergence is checked by $||$Vlocal_iter(ix)-Vlocal_iter-1(ix)$||^2$/(number of grids).

- **threshold (real(8), Default=1d-17 [a.u.] (for `convergence='rho_dne'`) and -1 (for other options of `convergence`))**

  Available for 'dft' and 'dft_md' options of `theory`.
  Threshold for convergence that is specified by `convergence` keyword.
  Unit conversions are: 1 a.u.= 45.54 $A^{-6}$for `convergence='norm_rho'` and `'norm_rho_dng'`, 1 a.u.= 33.72x$10^4$$A^{-6}$$eV^2$for `convergence='norm_pot'` and `'norm_pot_dng'`

- **nscf_init_redistribution (integer, Default=10)**

Available for 'dft' and 'dft_md' options of `theory`.

The number of initial iterations for redistribution of the occupation number in finite temperature calculation.

- **nscf_init_no_diagonal (integer, Default=10)**

  Available for `&scf/yn_subspace_diagonalization='y'` with 'dft' option of `theory`.

  The number of initial iterations for which subspace diagonalization is not done.

- **nscf_init_mix_zero (Integer, Default=-1)**

  Available for 'dft' option of `theory`.

  The densities is not mixed (i.e. fixed) during the given number of the SCF iteration cycle, that is, wavefunctions are optimized without updating the density.

- **conv_gap_mix_zero (real(8), Default=99999d0)**

  Available for positive number of `nscf_init_mix_zero` with 'dft' option of `theory`.

  The condition to quite the fixed density iteration forced by `step_initial_mix_zero` option. The density is allowed to start mixing after the band-gap energy exceeds this given gap threshold for consecutive five SCF iteration steps,

6.15 &emfield

- **trans_longi (character, Default='tr')**

  Available for `yn_periodic='y'` with 'maxwell' and TDDFT based options of `theory`.
  Boundary condition for fields on macro-scale in solid-state calculations.
  Options
  `'tr'` / Transverse
  `'lo'` / longitudinal

- **ae_shape1/ae_shape2 (character, Default='none')**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Envelope shape of the first/second pulse.
  Options
  `'impulse'` / Impulsive fields.
  `'Acos2'` / Envelope of $\cos^2$for a vector potential.
  `'Acos3'` / Envelope of $\cos^3$for a vector potential.
  `'Acos4'` / Envelope of $\cos^4$for a vector potential.
  `'Acos6'` / Envelope of $\cos^6$for a vector potential.
  `'Acos8'` / Envelope of $\cos^8$for a vector potential.
  `'Ecos2'` / Envelope of $\cos^2$for a electric field.
  `'Asin2cos'` [Trial] / Envelope of $\sin^2$with cosine type oscillation for a vector potential.
  `'Asin2_cw'` [Trial] / Envelope of $\sin^2$at beginning and continuous wave after that for a vector potential (for 'ae_shape1' only).
  `'input'` [Trial] / read-in user-defined field is used given by `file_input1` option (for 'ae_shape1' only).
  `'none'` / no incident field is applied

  For `yn_periodic='n'`, `'impulse'`, `'Acos2'`, and `'Ecos2'` can be chosen.

- **file_input1 (character, Default='')**

  Available for `theory='tddft_pulse'` with `ae_shape1='input'`.

  The input file name for user-defined incident field (vector potential) when `ae_shape1='input'` is used. The file must be numerical table (separated by blank) having more than four columns; the first column is time and second to fourth columns are Ax/c, Ay/c, Az/c, repsectively. All the quantities are written in units specified by `unit_system`, and '#' and '!' are available for a comment line. Besides, the linear interpolation is performed when the time step is differ from the calculation.

- **e_impulse (real(8), Default=1d-2 a.u.)**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Momentum of impulsive perturbation. This valiable has the dimention of momentum,
  energy*time/length.

- **E_amplitude1/E_amplitude2 (real(8), Default=0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Maximum amplitude of electric fields for the first/second pulse.

  This valiable has the dimension of electric field, energy/(length*charge). This cannot be set with `&emfield/I_wcm2_1` (`I_wcm2_2`) simultaneously.

- **I_wcm2_1/I_wcm2_2 (real(8), Default=-1d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Peak intensity (W/cm^2) of the first/second pulse.

  This valiable cannot be set with `&emfield/E_amplitude1` (`E_amplitude2`) simultaneously.

- **tw1/tw2 (real(8), Default=0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Duration of the first/second pulse (edge-to-edge time length).

- **omega1/omega2 (real(8), Default=0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.
  Mean photon energy (average frequency multiplied by the Planck constant) of the first/second pulse.

- **epdir_re1(3)/epdir_re2(3) (real(8), Default=1d0, 0d0, 0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.

  Real part of polarization unit vector for the first/second pulse.

- **epdir_im1(3)/epdir_im2(3) (real(8), Default=0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.

  Imaginary part of polarization unit vector for the first/second pulse.

- **phi_cep1/phi_cep2 (real(8), Default=0d0/0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.

  Carrier emvelope phase of the first/second pulse.

- **t1_t2 (real(8), Default=0d0)**

  Available for 'maxwell' and TDDFT based options of `theory`.

  Time-delay between the first and the second pulses.

- **t1_start (real(8), Default=0d0)**

Available for 'maxwell' and TDDFT based options of `theory`.

Time-delay of the first pulse. (this is not available for multiscale option).

- **num_dipole_source (integer, Default=0)**

  Available for TDDFT based options of `theory`.

  Number of radiation sources for exciting optical near fields as incident sources. Maximum number is `2`.

- **vec_dipole_source(3,num_dipole_source) (real(8), Default=0d0)**

  Available for TDDFT based options of `theory`.

  Dipole vectors of the radiation sources for exciting optical near fields as incident sources.

- **cood_dipole_source(3,num_dipole_source) (real(8), Default=0d0)**

  Available for TDDFT based options of `theory`.

  Central coordinates of the dipole vectors for exciting optical near fields as incident sources.

- **rad_dipole_diele (real(8), Default=2d0 a.u.)**

  Available for TDDFT based options of `theory`.

  Radii of dielectric spheres for exciting optical near fields as incident sources.

6.16 &singlescale

- **method_singlescale (character, Default='3d')**

  Available for `theory='single_scale_maxwell_tddft'`.
  Type of single-scale Maxwell-TDDFT method.
  Options:
  `'3d'` / 3-dimensional FDTD + TDDFT
  `'1d'` / 1-dimensional FDTD (along the z axis) + TDDFT
  `'1d_fourier'` / `'1d'` with 3D Fourier component of the vector potential

- **cutoff_G2_emfield (real(8), Default=-1d0)**

  Available for `theory='single_scale_maxwell_tddft'`.
  Cutoff energy of Fourier component of the vector potential when method_singlescale='1d_fourier'.

- **yn_symmetrized_stencil (character, Default='n')[Trial]**

  Available for `theory='single_scale_maxwell_tddft'`.
  Flag for the symmetrized finite differences of the product of the vector potential and the orbital wavefunction (nabla A(r) psi(r)).
  This option improves hermiticity of the Hamiltonian but makes worse the computational cost.

- **yn_put_wall_z_boundary (character, Default='n')[Trial]**

  Available for DFT/TDDFT based options of `theory`.
  Option to put potential wall on the boundary plane at z=0 and z=``&system/al(3)``. This is to prevent the electrons from crossing the z-boundary plane. In the single-scale + Maxwell method, the electron density on the z-boundary plane can make the norm conservation (of electrons) less accurate due to the discontinuity of the vectorpotential. The wall is given by the square of cosine function.
  Options
  `'y'` / put the potential wall

'n' / no potential wall

- **wall_height (real(8), Default=100.0 eV)**

  Available for yn_put_wall_z_boundary='y'.
  The height of the potential wall.

- **wall_width (real(8), Default=5.0 angstrom)**

  Available for yn_put_wall_z_boundary='y'.
  The width of the potential wall defined by the length from the potential peak (z=0 and z=‘‘&system/al(3)‘‘) to the edge.

6.17 &multiscale

- **fdtddim (character, Default='1d')[Trial]**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'
  Dimension of macroscopic scale system (Maxwell(FDTD) calculation) in multi-scale Maxwell-TDDFT method.
  Options:
  '3d' / 3-dimensional FDTD for macroscopic scale (currently not available)
  '1d' / 1-dimensional FDTD (along the x axis) for macroscopic scale

- **nx_m (integer, Default=1)**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'

  Number of macroscopic grid points inside materials for x-direction.

- **ny_m/nz_m (integer, Default=1)[Trial]**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'

  Number of macroscopic grid points inside materials for (y/z)-direction.

- **hx_m (real(8), Default=0d0)**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'

  Spacing of macroscopic grid points inside materials for (x)-direction. Unit of length can be chosen by &units/unit_length. Variable hx_m is deprecated, and will be moved to &units/dl_em(1)

- **hy_m/hz_m (real(8), Default=0d0)[Trial]**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'

  Spacing of macroscopic grid points inside materials for (y/z)-direction. Unit of length can be chosen by &units/unit_length. Variable hy_m and hz_m are deprecated, and will be moved to &units/dl_em(2:3)

- **nxvacl_m/nxvacr_m (integer, Default=1/0)**

  Available for theory='multi_scale_maxwell_tddft' with yn_periodic='y'

  Number of macroscopic grid points for vacumm region. nxvacl_m and nxvacr_m give the number for positive x-direction in front of material,

6.18 &maxwell

- **al_em(3) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Size of simulation box in electromagnetic analysis. Unit of the length can be chosen by `&units/unit_system`.

- **dl_em(3) (real(8), Default=0d0)**

  Available for `theory='maxwell'` and `theory='multi_scale_maxwell_tddft'`.

  Spacing of real-space grids in electromagnetic analysis. Unit of length can be chosen by `&units/unit_system`.

- **dt_em (real(8), Default=0)**

  Available for `theory='maxwell'`.

  Time step in electromagnetic analysis. Unit of time can be chosen by `&units/unit_system`.

- **nt_em (integer, Default=0)**

  Available for `theory='maxwell'`.

  Number of total time steps for real-time propagation in electromagnetic analysis.

- **boundary_em(3,2) (character, Default='default')**

  Available for `theory='maxwell'` and `theory='multi_scale_maxwell_tddft'`.

  Boundary condition in electromagnetic analysis. The first index(1-3 rows) corresponds to x, y, and z axes. The second index(1-2 columns) corresponds to bottom and top of the axes. If `&system/yn_periodic='n'`, `'default'`, `'abc'`, and `'pec'` can be chosen, where `'default'` automatically chooses `'abc'`. If `&system/yn_periodic='y'`, `'default'`, `'pml'`, and `'periodic'` can be chosen, where `'default'` automatically chooses `'periodic'`. `'abc'` is absorbing boundary, `'pec'` is perfect electric conductor, and `'periodic'` is periodic boundary.

- **shape_file (character, Default='none')**

  Available for `theory='maxwell'`.

  Name of shape file in electromagnetic analysis. The shape files can be generated by using SALMON utilities (https://salmon-tddft.jp/utilities.html).

- **media_num (integer, Default=0)**

  Available for `theory='maxwell'`.

  Number of media in electromagnetic analysis.

- **media_type(:) (character, Default='vacuum')**

  Available for `theory='maxwell'`.

  Type of media in electromagnetic analysis. `'vacuum'`, `'constant media'`, `'pec'`, and `'lorentz-drude'` can be chosen. If `'lorentz-drude'` is chosen, linear response calculation can be done by `&emfield/ae_shape1 or ae_shape2='impulse'`.

- **epsilon_em(:) (real(8), Default=1d0)**

  Available for `theory='maxwell'`.

  Relative permittivity of the media in electromagnetic analysis.

- **mu_em(:) (real(8), Default=1d0)**

Available for `theory='maxwell'`.

Relative permeability of the media in electromagnetic analysis.

- **sigma_em(:) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Conductivity of the media in electromagnetic analysis.

- **pole_num_ld(:) (integer, Default=1)**

  Available for `theory='maxwell'`.

  Number of poles of the media for the case of `type_media='lorentz-drude'` in electromagnetic analysis.

- **omega_p_ld(:) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Plasma frequency of the media for the case of `type_media='lorentz-drude'` in electromagnetic analysis.

- **f_ld(:,:) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Oscillator strength of the media for the case of `type_media='lorentz-drude'` in electromagnetic analysis. The first index is media id whose maximum value is determined by `media_num`. The second index is pole id whose maximum value is determined by `pole_num_ld`.

- **gamma_ld(:,:) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Collision frequency of the media for the case of `type_media='lorentz-drude'` in electromagnetic analysis. The first index is media id whose maximum value is determined by `media_num`. The second index is pole id whose maximum value is determined by `pole_num_ld`.

- **omega_ld(:,:) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Oscillator frequency of the media for the case of `type_media='lorentz-drude'` in electromagnetic analysis. The first index is media id whose maximum value is determined by `media_num`. The second index is pole id whose maximum value is determined by `pole_num_ld`.

- **wave_input (character, Default='none')**

  Available for `theory='maxwell'`.

  If `'source'`, the incident pulse in electromagnetic analysis is generated by the incident current source.

- **ek_dir1(3)/ek_dir2(3) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Propagation direction of the first/second pulse.

- **source_loc1(3)/source_loc2(3) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Location of the incident current source of the first/second pulse. Note that the coordinate system ranges from `-al_em/2` to `al_em/2` for `&system/yn_periodic='n'` while ranges from `0` to `al_em` for `&system/yn_periodic='y'`.

- **obs_num_em (integer, Default=0)**

  Available for `theory='maxwell'`.

  Number of observation point in electromagnetic analysis. From the obtained results, figure and animation files can be generated by using SALMON utilities (https://salmon-tddft.jp/utilities.html).

- **obs_samp_em (integer, Default=1)**

  Available for `theory='maxwell'`.

  Sampling time-step of the observation in electromagnetic analysis.

- **obs_loc_em(:,3) (real(8), Default=0d0)**

  Available for `theory='maxwell'`.

  Location of the observation point in electromagnetic analysis. Note that the coordinate system ranges from `-al_em/2` to `al_em/2` for `&system/yn_periodic='n'` while ranges from `0` to `al_em` for `&system/yn_periodic='y'`.

- **yn_obs_plane_em(:) (character, Default='n')**

  Available for `theory='maxwell'`.

  Enable(`'y'`)/disable(`'n'`). Output of the electrmagnetic fields on the planes (xy, yz, and xz planes) for each observation point. This option must be `'y'` for generating animation files by using SALMON utilities (https://salmon-tddft.jp/utilities.html).

- **yn_obs_plane_integral_em(:) (character, Default='n')**

  Available for `theory='maxwell'`.

  Enable(`'y'`)/disable(`'n'`). Output of the spatial integration of electrmagnetic fields on the planes (xy, yz, and xz planes) for each observation point.

- **yn_wf_em (character, Default='y')**

  Available for `theory='maxwell'`.

  Enable(`'y'`)/disable(`'n'`). Applying a window function for linear response calculation when `&calculation/theory=maxwell`.

6.19 &analysis

- **projection_option / out_projection_step (character/integer, Default='no'/100)[currently not available]**

  Available for TDDFT based options of `theory`.
  Methods of projection to analyze the excited states (e.g. the number of excited electrons.)
  Options
  `'no'` / no projection.
  `'gs'` / projection to eigenstates of ground-state Hamiltonian.
  `'rt'` / projection to eigenstates of instantaneous Hamiltonian.
  This is printed everty `out_projection_step` step during time-propagation.

- **nenergy (integer, Default=1000)** Number of energy grid points for frequency-domain analysis. This parameter is used, for examples, `theory='tddft_response'` and `theory='maxwell'`.

- **de (real(8), Default=0.01d0 eV)** Energy grid size for frequency-domain analysis. This parameter is used, for examples, `theory='tddft_response'` and `theory='maxwell'`.

- **out_rt_energy_step (integer, Default=10)**

Available for the TDDFT based option of `theory`.

Total energy is calculated and printed every `out_rt_energy_step` time steps.

- **yn_out_psi (character, Default='n')**

  Available for `theory='dft'`.
  Option for output of wavefunctions
  Options
  `'y'` / enable.
  `'n'` / disable.

  The format is specified by &analysis/`format_voxel_data`.

- **yn_out_dos (character, Default='n')**

  Available for `theory='dft'`.
  Option for output of density of state
  Options
  `'y'` / enable.
  `'n'` / disable.

- **yn_out_pdos (character, Default='n')**

  Available for `theory='dft'`.
  Option for output of projected density of state
  Options
  `'y'` / enable.
  `'n'` / disable.

- **yn_out_dos_set_fe_origin (character, Default='n')**

  Available for `yn_out_dos='y'` and `yn_out_pdos='y'`.
  Options to set the Fermi energy to zero
  `'y'` / enable
  `'n'` / disable.

  This option is not used if `&system/nstate` is equal to `&system/nelec`/2.

- **out_dos_start / out_dos_end (real(8), Default=-1d10 / 1d10 eV)**

  Available for `yn_out_dos='y'` and `yn_out_pdos='y'`.

  Lower/Upper bound (energy) of the density of state spectra. If this value is lower/higher than a specific value near the lowest/highest energy level, this parameter is re-set to the value.

- **out_dos_nenergy (integer, Default=601)**

  Available for `yn_out_dos='y'` and `yn_out_pdos='y'`.

  Number of energy points sampled in the density of state spectra.

- **out_dos_function (character, Default='gaussian')**

  Available for `yn_out_dos='y'` and `yn_out_pdos='y'`.
  Choise of smearing method for the density of state spectra.
  Options:
  `gaussian` / Gaussian function is used.
  `lorentzian` / Lorentzian function is used.

- **out_dos_width (real(8), Default=0.1d0 eV)**

Available for `yn_out_dos='y'` and `yn_out_pdos='y'`.

Smearing width used in the density of state spectra.

- **yn_out_dns (character, Default='n')**

  Available for `theory='dft'`.
  Option to print the spatial electron density distribution in the ground state.
  `'y'` / enable
  `'n'` / disable.

- **yn_out_dns_rt/out_dns_rt_step (Character/Integer, Default='n'/50)**

  Available for `theory='dft_md','tddft_pulse'`.
  Options to print the spatial electron density distribution everty `out_dns_rt_step` step during time-propagation.
  `'y'` / enable
  `'n'` / disable.

- **yn_out_dns_ac_je/out_dns_ac_je_step (Character/Integer, Default='n'/50)**

  Available for `theory='single_scale_maxwell_tddft'`.
  Options to print the electron density, vector potential, electronic current, and ionic coordinates every `outdns_dns_ac_je_step` time steps.
  `'y'` / enable
  `'n'` / disable.

  The data written in binary format are divided to files corresponding to the space-grid parallelization number.

- **yn_out_dns_trans/out_dns_trans_energy (Character/Real(8), Default='n'/1.55d0eV)[currently not available]**

  Available for `theory='tddft_pulse'`.
  Option to calculate transition in different density from the ground state at specified frequency omega(given by `out_dns_trans_energy`) by drho(r,omega)=FT(rho(r,t)-rho_gs(r))/T.
  `'y'` / enable
  `'n'` / disable.

  (currently not available)

- **yn_out_elf (character, Default='n')**

  Available for `theory='dft'`.
  Option to print the electron localization function.
  `'y'` / enable
  `'n'` / disable.

- **yn_out_elf_rt/out_elf_rt_step (Character/Integer,Default='n'/50)**

  Available for `theory='dft_md', 'tddft_pulse'`.
  Option to print the electron localization function during the time-propagation every `out_elf_rt_step` time steps.
  `'y'` / enable
  `'n'` / disable.

- **yn_out_estatic_rt/out_estatic_rt_step (Character/Integer, Default='n'/50)**

  Available for `theory='tddft_pulse'`.

Option to print the static electric field during the time-propagation every `out_estatic_rt_step` time steps.

`'y'` / enable

`'n'` / disable.

- **yn_out_rvf_rt/out_rvf_rt_step (Character/Integer, Default='n'/10)**

  Available for TDDFT based options and 'dft_md' option of `theory`.

  Option to print the coordinates[A], velocities[au], forces[au] on atoms during time-propagation in `SYSname`_trj.xyz every `out_rvf_rt_step` time steps.

  `'y'` / enable

  `'n'` / disable.

  If `yn_md='y'`, the printing option is automatically turned on.

- **yn_out_tm (character, Default='n')[Trial]**

  Available for `yn_periodic='y'` with `theory='dft'`.

  Option to calculate and print the transition moments between occupied and virtual orbitals to `SYSname`_tm.data after the ground state calculation.

  `'y'` / enable

  `'n'` / disable.

- **out_ms_step (integer, Default=100)**

  Available for `theory='multi_scale_maxwell_tddft'`.

  Option to print some information every `out_ms_step` time step in the Maxwell + TDDFT multi-scale calculation.

- **format_voxel_data (character, Default='cube')**

  Available for `yn_out_psi='y'`, `yn_out_dns(_rt)='y'`, `yn_out_dns_ac_je='y'`, `yn_out_elf(_rt)='y'`, `yn_out_estatic_rt='y'`.

  Option of the file format for three-dimensional volumetric data.

  `'avs'` / AVS format

  `'cube'` / cube format

  `'vtk'` / vtk format

- **nsplit_voxel_data (integer, Default=1)**

  Available for `format_voxel_data='avs'`.

  Number of separated files for three dimensional data.

- **yn_out_perflog (character(1), Default='y')**

  Available for all `theory`

  Option to print the performance log of routines and modules.

- **format_perflog (character(6), Default='stdout')**

  Available for `yn_out_perflog = 'y'`

  The output format of performance log.

  `'stdout'` / standard output unit

  `'text'` / save to text file

  `'csv'` / save to csv format file

6.20 &poisson

- **layout_multipole (character, Default=3)**

  Available for `yn_periodic='n'` with DFT and TDDFT based options of `theory`.

  A variable to determine how to put multipoles in the Hartree potential calculation.

  Options:

  `1`/ A single pole is put at the center.
  `2`/ Multipoles are put at the center of atoms.
  `3`/ Multipoles are put at the center of mass of electrons in prepared cuboids.

- **num_multipole_xyz(3) (integer, Default=0)**

  Available for `yn_periodic='n'` with DFT and TDDFT based options of `theory`.
  Number of multipoles. When default is set, number of multipoles is calculated automatically.

- **lmax_multipole (integer, Default=4)[Trial]**

  Available for `yn_periodic='n'` with DFT and TDDFT based options of `theory`.
  A maximum angular momentum for multipole expansion in the Hartree-cg calculation.

- **threshold_cg (real(8), Default=1d-15 a.u.(= 1.10d-13 A^3eV^2))**

  Available for `yn_periodic='n'` with DFT and TDDFT based options of `theory`.
  A convergence value for the Hartree-cg calculation. The convergence is checked by ||tVh(i)-tVh(i-1)||^2/(number of grids).

6.21 &ewald

- **newald (integer, Default=4)**

  Available for `yn_periodic='y'` with DFT/TDDFT based options of `theory`.
  Parameter for Ewald method for ion-ion Coulombic interaction. Short-range part of Ewald sum is calculated within `newald` th nearlist neighbor cells.

- **aewald (real(8), Default=0.5d0)**

  Available for `yn_periodic='y'` with DFT/TDDFT based options of `theory`.

  Square of range separation parameter for Ewald method in atomic unit.

- **cutoff_r (real(8), Default=-1d0)**

  Available for `yn_periodic='y'` with DFT/TDDFT based options of `theory`.
  Cut-off length in real-space. This is automatically chosen in default (negative number)

- **cutoff_r_buff (real(8), Default=2d0 a.u.)**

  Available for `yn_periodic='y'` with `yn_md='y'` or `theory='dft_md'`.
  Buffer length in radius for book-keeping for real-space interaction.

- **cutoff_g (real(8), Default=-1d0)**

  Available for `yn_periodic='y'` with DFT/TDDFT based options of `theory`.
  Cut-off in G-space in the Ewald method. No cut-off in default.

6.22 &opt[Trial]

- **nopt (integer, Default=100)**

  Available for `yn_opt='y'` with `theory='dft'`.

  The maximum step number of geometry optimization.

- **convrg_opt_fmax (real(8), Default=1d-3 [a.u.])**

  Available for `yn_opt='y'` with `theory='dft'`.
  Convergence threshold of geometry optimization in maximum force on atom.

- **max_step_len_adjust (real(8), Default=-1d0)**

  Available for `yn_opt='y'` with `theory='dft'`.
  Set maximum optimization step length (if positive number is given)

6.23 &md[Trial]

- **ensemble (character, Default='NVE')**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Ensemble in MD option:
  Options:
  `NVE`/ NVE ensemble (constant energy and volume system)
  `NVT`/ NVT ensemble (constant temperature and volume system)

- **thermostat (character, Default='nose-hoover')**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Thermostat in "NVT" option:
  Options:
  `nose-hoover`/ Nose-Hoover thermostat.

- **step_velocity_scaling (integer, Default=-1)**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Time step interval for velocity-scaling. Velocity-scaling is applied if this is set to positive.

- **step_update_ps (Integer, Default=10)**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Time step interval for updating pseudopotential (Larger number makes calculation time reduce but gets inaccurate).

- **temperature0_ion_k (real(8), Default=298.15d0 [K])**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Setting ionic temperature [K] for NVT ensemble, velocity scaling and generating initial velocities.

- **yn_set_ini_velocity (character, Default='n')**

  Available for `yn_md='y'` or `theory='dft_md'`.
  Option to generate initial velocities.
  Options:
  `y`/ Generate initial velocity with Maxwell-Bortzman distribution.
  `n`/ disable.

- **file_ini_velocity (character, Default='none')[Trial]**

  Available for `yn_md='y'` or `theory='dft_md'`.

  File name for reading initial velocities. This is read if the file name is given, then, the priority is higher than use of `set_ini_velocity` and restart data of velocities. The format is simply vx(iatom) vy(iatom) vz(iatom) in each line. The order of atoms must be the same as the given coordinates in the main input file. In case of using nose-hoover thermostat, a thermostat variable should be put at the last line (all atomic unit).

- **thermostat_tau (real(8), Default=41.34d0 a.u. or 1d0 fs)**

  Available for `yn_md='y'` or `theory='dft_md'`.

  Parameter in Nose-Hoover method: controlling time constant for temperature.

- **yn_stop_system_momt (character, Default='n')**

  Available for `yn_md='y'` or `theory='dft_md'`.

  Center of mass is fixed every time step.

  Options:

  `y`/ enable.

  `n`/ disable.

6.24 &code

- **yn_want_stencil_hand_vectorization (character, Default='y')**

  This option requests hand-vectorized optimization code of stencil in the hamiltonian calculation.

  SALMON checks the calculation can be used the hand-vectorized code.

  If failing it, SALMON will uses the typical implementation.

- **yn_want_communication_overlapping (character, Default='n')**

  Available for `theory='tddft*' or '*maxwell_tddft'`

  This option requests computation/communication overlap algorithm to improve the performance of stencil in the hamiltonian calculation.

  SALMON checks the calculation can be used the overlap algorithm.

  If failing it, SALMON will uses the non-overlap algorithm.

- **stencil_openmp_mode (character, Default='auto')**

  This option selects a OpenMP parallelization mode of stencil in the hamiltonian calculation.

  `auto` / SALMON decides the parallelization target automatically.

  `orbital` / OpenMP parallelization is applied to orbital (and k-point) loop.

  `rgrid` / OpenMP parallelization is applied to real-space grid loop.

- **current_openmp_mode (character, Default='auto')**

  This option selects a OpenMP parallelization mode of the current calculation.

  `auto` / SALMON decides the parallelization target automatically.

  `orbital` / OpenMP parallelization is applied to orbital (and k-point) loop.

  `rgrid` / OpenMP parallelization is applied to real-space grid loop.

- **force_openmp_mode (character, Default='auto')**

  This option selects a OpenMP parallelization mode of the force calculation.

  `auto` / SALMON decides the parallelization target automatically.

`orbital` / OpenMP parallelization is applied to orbital (and k-point) loop.
`rgrid` / OpenMP parallelization is applied to real-space grid loop.

BIBLIOGRAPHY

[1] M. Noda, S. A. Sato, Y. Hirokawa, M. Uemoto, T. Takeuchi, S. Yamada, A. Yamada, Y. Shinohara, M. Yamaguchi, K. Iida, I. Floss, T. Otobe, K.-M. Lee, K. Ishimura, T. Boku, G. F. Bertsch, K. Nobusada, and K. Yabana. Salmon: scalable ab-initio light-matter simulator for optics and nanoscience. *Comp. Phys. Comm.*, 235(356-365):, 2019.

[2] Masashi Noda, Kazuya Ishimura, Katsuyuki Nobusada, Kazuhiro Yabana, and Taisuke Boku. Massively-parallel electron dynamics calculations in real-time and real-space: toward applications to nanostructures of more than ten-nanometers in size. *Journal of Computational Physics*, 265:145–155, 2014.

[3] George F Bertsch, J-I Iwata, Angel Rubio, and Kazuhiro Yabana. Real-space, real-time method for the dielectric function. *Physical Review B*, 62(12):7998, 2000.

[4] K. Yabana and G. F. Bertsch. Time-dependent local-density approximation in real time. *Phys. Rev. B*, 54:4484–4487, Aug 1996. URL: https://link.aps.org/doi/10.1103/PhysRevB.54.4484, doi:10.1103/PhysRevB.54.4484.

[5] Kazuhiro Yabana, T Sugiyama, Y Shinohara, T Otobe, and GF Bertsch. Time-dependent density functional theory for strong electromagnetic fields in crystalline solids. *Physical Review B*, 85(4):045134, 2012.

[6] Shunsuke A. Sato and Kazuhiro Yabana. Maxwell + tddft multi-scale simulation for laser-matter interactions. *Journal of Advanced Simulation in Science and Engineering*, 1(1):98–110, 2014. doi:10.15748/jasse.1.98.

[7] Yuta Hirokawa, Taisuke Boku, Shunsuke A Sato, and Kazuhiro Yabana. Electron dynamics simulation with time-dependent density functional theory on large scale symmetric mode xeon phi cluster. In *Parallel and Distributed Processing Symposium Workshops, 2016 IEEE International*, 1202–1211. IEEE, 2016.