SALMON document

Release v.2.2.2

SALMON developers

Jun 14, 2025

Contents

Jun 14, 2025

SALMON (Scalable Ab initio Light-Matter simulator for Optics and Nanoscience) 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 1

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

The development of SALMON is in progress at GitHub.com

- SALMON2: Development version of SALMON.

- SALMON-DOCS: Manual for the development version.
- SALMON-inputs: Database for SALMON input files used in published papers.

1.5 List of developers

(Alphabetic order)

- Kenji Iida (Hokkaido University, Japan)
- Masashi Noda (Academeia, Japan)
- Tomohito Otobe (National Institutes for Quantum Science and Technology, Japan)
- Shunsuke Sato (Tohoku University, Japan)
- Yasushi Shinohara (NTT, Japan)
- Mizuki Tani (National Institutes for Quantum Science and Technology, Japan)
- Mitsuharu Uemoto (Kobe University, Japan)
- Kazuhiro Yabana (University of Tsukuba, Japan)
- Atsushi Yamada (National Defense Academy of Japan, Japan)
- Shunsuke Yamada (National Institutes for Quantum Science and Technology, Japan)

1.6 Former developers

- Isabella Floss
- Yuta Hirokawa
- Jun-Ichi Iwata
- Yuki Ito
- Kazuya Ishimura
- Kyung-Min Lee
- Katsuyuki Nobusada
- Takashi Takeuchi
- Xiao-Min Tong
- Maiku Yamaguchi

1.7 How to cite SALMON

1.7.1 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:

CHAPTER 2

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)
  - Nvidia HPC SDK Compiler
- 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 specify them manually or customize the configuration files for CMake (see *Additional options in configure.py script*). If no numerical libraries are installed on your system, the BLAS/LAPACK package will be automatically downloaded and built during the compilation process.

For installing SALMON, we recommend using CMake as the primary method. If you encounter any issues using CMake in your environment, you may use GNU Make as an alternative. If you run into problems during the build process, refer to *Troubleshooting of the Installation Process*.

2.2 Download

The latest version of SALMON can be downloaded from download page. You can also download the file using the following command:

```
$ wget http://salmon-tddft.jp/download/SALMON-<VERSION>.tar.gz
```

To extract the contents of the downloaded file `SALMON-<VERSION>.tar.gz`, use the following command:

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

After extraction, the following directories will be created:

```
SALMON
  |- src          Source codes
  |- samples      Sample input files
  |- cmakefiles   CMake related files
  |- platforms    CMake configuration files
  |- gnumakefiles GNU Makefiles for building
  | ...
```

2.3 Build and Install

To compile SALMON and create the executable binary, we recommend using CMake as the primary method. If you are unable to build SALMON with CMake in your environment, you may use GNU Make as an alternative (*Build using GNU Makefile*).

2.3.1 Checking CMake availability

First, check whether CMake is available in your environment. Type the following command in a Linux terminal:

```
$ 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

After confirming that CMake version 3.14.0 or later is available in your environment, proceed with the following steps. We assume that you are currently in the `SALMON` directory.

1. Create a new temporary directory named `build` and move into it:

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

2. Run the Python script `configure.py`, then build and install SALMON:

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

(Replace `INSTALLATION_DIRECTORY` with your desired installation directory. If this is not specified, the executable file will be created in the `build` directory.)

When executing the Python script, you need to specify an `ARCHITECTURE` that represents the CPU architecture of your computer system, such as `intel-avx512`. The main options for `ARCHITECTURE` are as follows:

arch	Detail	Compiler	Numerical Library
intel-oneapi	Intel oneAPI (cross-architecture)	Intel Compiler	Intel MKL
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
nvhpc-openmp	Nvidia OpenMP (CPU)	Nvidia HPC Compiler	Nvidia HPC SDK
nvhpc-openacc	Nvidia OpenACC (GPU)	Nvidia HPC Compiler	Nvidia HPC SDK
nvhpc-openacc-cuda	Nvidia OpenACC+CUDA (GPU)	Nvidia HPC Compiler	Nvidia HPC SDK

If there is no suitable option, you can customize a CMake configuration file or specify compilers and flags manually (See *Additional options in configure.py script*). If the build completes successfully, an executable file named `salmon` will be created in the `INSTALLATION_DIRECTORY`.

2.4 Files necessary to run SALMON

To run SALMON, at least two types of files are required for any calculations. One is a text file containing input variables of SALMON (the SALMON input file), which should be read from standard input (`stdin`). This file must be prepared in Fortran90 namelist format. Pseudopotential files for the relevant elements are also required. Depending on your purpose, additional files may be needed. For example, the atomic coordinates of the target material can either be written in the input file or provided in a separate file.

2.4.1 Pseudopotentials

SALMON utilizes the norm-conserving (NC) pseudpotential. Filenames of pseudopotentials should be written in the input file.

You can find pseudopotentials for some elements in the sample files provided in *Exercises*. SALMON supports several formats of pseudopotentials, as listed below. For example, pseudopotentials with the `.fhi` extension can be obtained from the ABINIT website; these are part of the older atomic data files used by the ABINIT code.

Pseudopotential	extension	Website
Fritz-Haber-Institute (FHI) pseudopotentials	`.fhi`	https://abinit.github.io/abinit_web/ATOMICDATA/LDA_FHI.zip (for LDA), https://abinit.github.io/abinit_web/ATOMICDATA/fhi.zip (for GGA)
Pseudopotentials for the OpenMX code	`.vps`	https://www.openmx-square.org/vps_pao2019/
Format 8 for ABINIT norm-conserving pseudopotentials	`.psp8`	https://abinit.github.io/abinit_web/pseudopotential.html , http://www.pseudo-dojo.org/
Unified-pseudopotential-format (NC type only in SALMON)	`.upf`	http://pseudopotentials.quantum-espresso.org/home/unified-pseudopotential-format , http://www.pseudo-dojo.org/

2.4.2 SALMON input file

The SALMON input file consists of several blocks of namelists, as shown below:

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

A block of namelists starts with a line beginning with `&` and ends with a line containing only `/`. These blocks may appear in any order.

Between the `&` and `/` lines, variables and their corresponding values are described. Many variables have default values, so it is not necessary to specify all of them. Variable definitions can appear in any order within the block.

Input variables are either integer, real (`real(8)`), or string (`character`) types. Some variables are arrays. A variable beginning with `yn_` is a string variable whose value is either `'y'` or `'n'` (i.e., yes or no). All string variables are case-insensitive.

SALMON simulates electron dynamics in systems with either isolated or periodic boundary conditions. The boundary condition is specified by the variable `yn_periodic` in the `&system` namelist.

Calculations are generally performed in two steps: first, a ground-state calculation is carried out, followed by a real-time electron dynamics simulation. The calculation mode or theory is specified by the variable `theory` in the `&calculation` namelist. Typically, a ground-state calculation based on DFT is performed by setting `theory = 'dft'`. Then, a real-time electron dynamics calculation based on TDDFT is carried out by setting `theory = 'tddft_pulse'`.

In *Exercises*, we provide six exercises that cover typical calculations feasible with SALMON. We also provide explanations of the input files used in these exercises, which can help you prepare input files for your own purposes. Additional examples of input files can be found in the SALMON-inputs database.

There are more than 20 groups of namelists. A complete list of namelist variables is given in *List of input keywords*.

2.5 Run SALMON

Before running SALMON, the following preparations are required, as described above: the salmon executable must be built from the source code, and both an input file (for example, `inputfile.inp`) and pseudopotential files must be prepared.

A calculation can be executed as follows:

In a single-process environment, type the following command:

```
$ salmon < inputfile.inp > stdout.log
```

(Here, it is assumed that the environment variable `$PATH` is properly set to include the SALMON executable.)

In a multi-process environment, where the command for parallel execution via MPI is `mpiexec`, use the following:

```
$ mpiexec -n NPROC salmon < inputfile.inp > stdout.log
```

Here, `NPROC` is the number of MPI processes to be used.

The execution command and job submission procedure may vary depending on the local environment. Below is a general summary of the conditions for running SALMON:

- SALMON runs in both single-process and multi-process (MPI) environments.
- The executable file is named `salmon` in the standard build process.
- To begin a calculation, a input file must be provided via `stdin`.

2.5.1 MPI process distribution

SALMON provides three variables to control process distribution and allocation.

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

By default, SALMON automatically determines the process distribution. However, in many cases, explicitly specifying the process distribution can result in better performance than relying on the default settings.

We recommend the following strategy for process distribution:

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 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 supercomputers. Therefore, the following contents will only be useful only for limited users.

2.6.1 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 *Additional options in configure.py script*.

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
/
```

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.

2.6.2 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 routine, 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'
/
```

2.6.3 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.
/
```

2.6.4 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'`

2.6.5 GPU acceleration

GPU acceleration (OpenACC or OpenACC+CUDA) for the DFT/TDDFT computation is available. For compiling SALMON for GPUs, specify `--arch=nvhpc-openacc` (OpenACC, recommended) or `--arch=nvhpc-openacc-cuda` (OpenACC+CUDA) option when executing `configure.py`. This option is currently under development and tested only for NVIDIA HPC SDK compiler with NVIDIA GPUs.

Note: Currently, the performance of the TDDFT part is well-tuned but the DFT part is not. We recommend executing DFT (ground-state) calculations on CPUs and TDDFT calculations on GPUs.

Multi-GPU run

For MPI calculations with multiple GPUs, the assignment of MPI processes to GPUs via `CUDA_VISIBLE_DEVICES` and the use of `nvidia-cuda-mps-control` can improve the performance of SALMON. The following example is a wrapper script for using those:

```
$ cat wrapper.sh
#! /bin/bash
### wrapper.sh
NCUDA_GPUS=${NCUDA_GPUS:-`nvidia-smi -L | wc -l`}
if [[ $OMPI_COMM_WORLD_LOCAL_SIZE -gt $NCUDA_GPUS ]]
then
  if [[ $OMPI_COMM_WORLD_LOCAL_RANK -eq 0 ]]
  then
    nvidia-cuda-mps-control -d
  fi
  sleep 10
fi
export CUDA_VISIBLE_DEVICES=$((${OMPI_COMM_WORLD_LOCAL_RANK} % ${NCUDA_GPUS}))
exec $@
if [[ $OMPI_COMM_WORLD_LOCAL_SIZE -gt $NCUDA_GPUS ]]
then
  echo quit | nvidia-cuda-mps-control
fi
```

Here, we used environment variables of OpenMPI, such as `$OMPI_COMM_WORLD_LOCAL_SIZE`. For MPI execution, use the following command:

```
$ mpirun -np ${num_MPI_processes} -npernode ${num_MPI_processes_per_node} \
   wrapper.sh ${program} < ${input} > stdout.log
```

Here, ${program} is the path of SALMON, ${input} is the input file, etc.

2.7 Troubleshooting of the Installation Process

2.7.1 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
```

2.8 Appendix

2.8.1 Additional options in configure.py script

Manual specifications of compiler and environment variables

When executing `configure.py`, users can specify several options and environment variables.

A list of available options for `configure.py` can be displayed with the following command:

```
$ python ../configure.py --help
```

The main options 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
LDFLAGS	linker flags

Using these options, you can manually specify the compilers and flags instead of using the `--arch` option. For example:

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

Customize CMake file for `--arch`

Users can find several CMake configuration files corresponding to the `--arch` options in the `platforms/` directory. If there is no suitable configuration file, you can copy one of the existing ones and customize it for your environment. For example, if there is a configuration file named `example.cmake` in the `platforms/` directory, the `configure.py` script can read it using the following command:

```
$ python ../configure.py --arch=example
```

Required libraries

In the build procedure of SALMON, CMake searches the following libraries. If the libraries are not found in the path specified by environment variables, the required libraries will be downloaded and compiled automatically.

- BLAS/LAPACK
  - Required by default compilation.
  - Most math libraries include BLAS/LAPACK by default.
  - `--with-lapack`: Path specification.
  - If the library is not found, it will be automatically downloaded from http://www.netlib.org/lapack/
- ScaLAPACK

  - Required by `--enable-scalapack`.
  - `--with-lapack`: Path specification.
  - If the library is not found, it will be automatically downloaded from http://www.netlib.org/scalapack/
- Libxc
  - Required by `--enable-libxc`.
  - `--with-libxc`: Path specification.
  - If the path is unspecified, the library will be automatically downloaded from https://libxc.gitlab.io
- EigenExa
  - Required by `--enable-eigenexa`. (`--enable-scalapack` is also required for EigenExa.)
  - EigenExa will be downloaded and built automatically even if the library is installed on your machine.
  - Automatically download from https://www.r-ccs.riken.jp/labs/lpnctrt/assets/img/EigenExa-2.4b.tgz

Build for single process calculations

When using the `--arch` option, MPI parallelization is enabled as default. 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 GNU Compiler Collection (GCC)

The architecture option `--arch` does not support GNU Compiler Collection (GCC). If you want to build SALMON by GCC, specify `FC` and `CC` as follows:

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

Here, `--enable-mpi` is required for the MPI parallelization. Note that the MPI parallelization is disabled as default when `--arch` option is not used. Compiler options can also be specified by `FFLAGS` and `CFLAGS`. For GCC 10 or later versions, `FFLAGS="-fallow-argument-mismatch"` may be required.

Compilation examples

Some compilation (configure) examples in several environments are shown below.

- Wisteria-Odyssey (University of Tokyo) & Fujitsu compiler, compiling with EigenExa:

```
$ python3 ../configure.py --arch=fujitsu-a64fx-ea --enable-scalapack --enable-
↪eigenexa FFLAGS="-fPIC"
```

- Cygnus (GPU supercomputer @ University of Tsukuba) & NVidia HPC SDK compiler version 23.11:

```
$ module load cmake/3.18.6 openmpi/nvhpc/23.11
$ python3 ../configure.py --arch=nvhpc-openacc LDFLAGS=-L/system/apps/nvhpc/23.11/
↪Linux_x86_64/23.11/math_libs/lib64/
```

- AWS Graviton2 machine (Amazon EC2 T4g instance) & Arm compiler:

```
$ python3 ../configure.py FC=armflang CC=armclang FFLAGS="-armpl" CFLAGS="-armpl"
```

- MacOS & GCC version 11:

```
$ brew install gcc@11
$ export FC=/opt/homebrew/Cellar/gcc@11/11.5.0/bin/gfortran-11
$ export CC=/opt/homebrew/Cellar/gcc@11/11.5.0/bin/gcc-11
$ export CXX=/opt/homebrew/Cellar/gcc@11/11.5.0/bin/g++-11
$ python ../configure.py FFLAGS="-fallow-argument-mismatch"
```

2.8.2 Compilation with FFTW library

For solving the Poisson equation for the Hartree potential, SALMON uses the discrete Fourier transform. FFTW library (https://www.fftw.org) is available for fast calculation. When executing `configure.py`, specify `--enable-fftw` option and linker flags for FFTW such as `LDFLAGS="-lfftw3_mpi -lfftw3"`.

Exapmle:

```
$ python ../configure.py --arch=ARCHITECTURE --enable-fftw LDFLAGS="-lfftw3_mpi -
↪lfftw3"
```

2.8.3 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/`.

CHAPTER 3

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 11 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 using a static density functional theory (DFT). This is illustrated in *Exercise-1*. After finishing the ground state calculation, two exercises of electron dynamics calculations based on time-dependent density functional theory (TDDFT) 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 calculation of the crystalline silicon based on DFT. *Exercise-5* illustrates the calculation of linear response properties of the crystalline silicon based on TDDFT 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 a simultaneous calculation of the propagation of a pulsed light and electronic motion in a bulk silicon, coupling Maxwell equations for the electromagnetic fields of the pulsed light and the electron dynamics in the unit cells based on TDDFT. This calculation is quite time-consuming and is recommended to execute using massively

parallel supercomputers. *Exercise-7* illustrates the calculation of a linearly polarized pulsed light irradiating normally on a surface of a bulk silicon.

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

Next 2 exercises (Exercise-10 ~ 11) are for a macroscopic light propagation through a metallic nanosphere solving Maxwell equations. The optical response of the nanoparticle is described by a dielectric function. Finite-Difference Time-Domain (FDTD) method is used to calculated the three-dimensional light propagation. *Exercise-10* illustrates the calculation of absorption-, scattering-, and extinction-cross-sections for an isolated Au nanoparticle. *Exercise-11* illustrates the calculation of absorption-, reflection-, and transmission-rates for a metasurface in which Au nanoparticles are periodically arrayed in two-dimension.

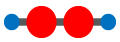
Input files of exercises are included in SALMON, in the directory `SALMON/samples/exercise_##_<description>/`.

3.2 C2H2 (isolated molecules)

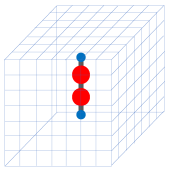
3.2.1 Exercise-1: Ground state of C2H2 molecule

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.

Acetylene molecule is a linear chain molecule composed of two Carbon atoms and two Hydrogen atoms.



In SALMON, we use a three-dimensional (3D) uniform grid system to express physical quantities such as electron orbitals.

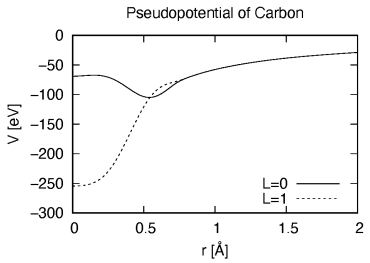


Input files

To run the code, following files in the directory `SALMON/samples/exercise_01_C2H2_gs/` 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

Pseudopotential files are needed for two elements, Carbon (C) and Hydrogen (H). The pseudopoential depends on the angular momentum, and looks as follows (for Carbon).



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 input keywords*.

```
!################################################################################################
↪###!
! Excercise 01: Ground state of C2H2 molecule                                                    ␣
↪    !
!------------------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                                     ␣
↪    !
!------------------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                                         ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                                           ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                                 ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                                 ␣
↪    !
!################################################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

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

yn_periodic specifies whether or not periodic boundary condition is applied.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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. !
  !------------------------------------------------!
/
```

file_pseudo(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 64, 64, 64
/
```

dl(i) specifies the spatial grid spacing in i-th direction.
num_rgrid(i) specifies the number of grid points in i-th direction.

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

nscf specifies the maximum number of SCF iterations.
threshold specifies the threshold to judge the convergence.

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

yn_out_psi, *yn_out_dns*, *yn_out_dos*, *yn_out_pdos*, *yn_out_elf* specify output files that are generated after the calculation.

```
&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) !
  !-------------------------------------------------------------!
/
```

&atomic_coor specifies spatial coordinates of atoms.

Execusion

In a multiprocess environment, calculation will be executed as:

```
$ mpiexec -n NPROC salmon < C2H2_gs.inp > C2H2_gs.out
```

where NPROC is the number of MPI processes. A standard output will be stored in the file `C2H2_gs.out`.

Output files

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

name	description
C2H2_info.data	information on ground state solution
C2H2_eigen.data	orbital 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_1/exercise_zip_files/01_C2H2_gs.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###############################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                              Version 2.0.1
```

(continues on next page)

(continued from previous page)

```
#
################################################################################
  Libxc: [disabled]
   theory= dft
   use of real value orbitals =  T
 ======
 MPI distribution:
   nproc_k      :           1
   nproc_ob     :           1
   nproc_rgrid :           1           1           2
 OpenMP parallelization:
   number of threads :         256
 .........
```

After that, the SCF loop starts. At each iteration step, the total energy as well as orbital energies and some other quantities are displayed.

```
-----------------------------------------------
iter=      1      Total Energy=       -197.59254070      Gap=    -20.17834599      Vh iter=␣
↪234
    1        -29.9707       2        -28.3380       3        -13.0123       4          5.8457
    5         -9.9213       6        -14.3326
iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec         =       1 0.31853198E+00
Ne=    10.0000000000000
-----------------------------------------------
iter=      2      Total Energy=       -280.97950515      Gap=     -9.59770609      Vh iter=␣
↪247
    1        -17.4334       2        -24.4941       3        -20.1872       4          0.8020
    5         -3.4058       6         -8.7957
iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec         =       2 0.54493263E+00
Ne=    10.0000000000000
-----------------------------------------------
iter=      3      Total Energy=       -295.67034640      Gap=     -6.90359156      Vh iter=␣
↪229
    1        -16.0251       2        -19.7759       3        -17.6765       4         -0.9015
    5         -2.9323       6         -7.8050
iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec         =       3 0.13010987E+00
Ne=    10.0000000000000
```

When the convergence criterion is satisfied, the SCF calculation ends.

```
-----------------------------------------------
iter=    162      Total Energy=       -339.69525272      Gap=      6.78870999      Vh iter=␣
↪  1
    1        -18.4106       2        -13.9966       3        -12.4163       4         -7.3386
    5         -7.3386       6         -0.5498
iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec         =     162 0.50237787E-08
Ne=    9.99999999999999
-----------------------------------------------
iter=    163      Total Energy=       -339.69525269      Gap=      6.78870999      Vh iter=␣
↪  1
    1        -18.4106       2        -13.9966       3        -12.4163       4         -7.3386
    5         -7.3386       6         -0.5498
iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec         =     163 0.69880308E-09
Ne=    9.99999999999999
 #GS converged at    164   : 0.69880308E-09
```

Next, the force acting on ions and some other information related to orbital energies are shown.

```
===== force =====
    1 -0.33652081E-05  0.16854696E-04 -0.59496450E+00
    2 -0.59222259E-06  0.24915590E-05  0.57651725E+00
    3 -0.37839836E-05  0.20304090E-04  0.59493028E+00
    4 -0.86779607E-06  0.39560274E-05 -0.57651738E+00
orbital energy information------------------------------
Lowest occupied orbital -0.676576619015730
Highest occupied orbital (HOMO) -0.269686750876529
Lowest unoccupied orbital (LUMO) -2.020624936948345E-002
Highest unoccupied orbital -2.020624936948345E-002
HOMO-LUMO gap  0.249480501507045
Physicaly upper bound of eps(omega)  0.656370369646246
------------------------------------------------------------
Lowest occupied orbital[eV]  -18.4105868958642
Highest occupied orbital (HOMO)[eV]  -7.33855002098465
Lowest unoccupied orbital (LUMO)[eV] -0.549840032009334
Highest unoccupied orbital[eV] -0.549840032009334
HOMO-LUMO gap[eV]   6.78870998897532
Physicaly upper bound of eps(omega)[eV]   17.8607468638548
------------------------------------------------------------
 writing restart data...
 writing completed.
```

In the directory `data_for_restart`, files that will be used in the next-step time evolution calculations are stored.

Other output files include following information.

C2H2_info.data

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

C2H2_eigen.data

Orbital 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`.

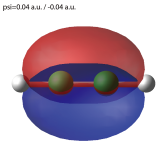
elf.cube

A cube file for electron localization function (ELF).

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

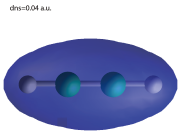
- **Highest occupied molecular orbital (HOMO)**

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



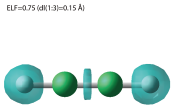
- **Electron density**

  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*.

Polarizability $\alpha_{\mu\nu}(t)$ is the basic quantity that characterizes optical responses of molecules and nano-particles, where μ, ν indicate Cartesian components, $\mu, \nu = x, y, z$. The polarizability $\alpha_{\mu\nu}(t)$ relates the μ component of the electric dipole moment at time t, $p_\mu(t)$, with the ν component of the electric field at time t',

$p_\mu(t) = \sum_{\nu=x,y,z} \alpha_{\mu\nu}(t-t')E_\nu(t')$.

We introduce a frequency-dependent polarizability by the time-frequency Fourier transformation of the polarizability,

$\tilde{\alpha}_{\mu\nu}(\omega) = \int dt e^{i\omega t}\alpha_{\mu\nu}(t)$.

The imaginary part of the frequency-dependent polarizability is related to the photoabsorption cross section $\sigma(\omega)$ by

$\sigma(\omega) = \frac{4\pi\omega}{c}\frac{1}{3}\sum_{\mu=x,y,z} \mathrm{Im}\tilde{\alpha}_{\mu\mu}(\omega)$.

The photoabsorption cross section is also related to the oscillator strength distribution by

$\sigma(\omega) = \frac{2\pi^2 e^2}{mc}\frac{df(\omega)}{d\omega}$.

In SALMON, the polarizability is calculated in time domain. First the ground state orbital $\phi_i(\mathbf{r})$ that satisfies the Kohn-Sham equation,

$H_{\rm KS}\phi_i(\mathbf{r}) = \epsilon_i\phi_i(\mathbf{r})$,

is prepared. Then an impulsive force given by the potential

$V_{\rm ext}(\mathbf{r},t) = I\delta(t)z$,

is applied to all electrons in the C2H2 molecule along the molecular axis which we take z axis. I is the magnitude of the impulse, and $\delta(t)$ is the Dirac's delta function. The orbital is distorted by the impulsive force at $t = 0$. Immediately after the impulse is applied, the orbital becomes

$\psi_i(\mathbf{r}, t = 0_+) = e^{iIz/\hbar}\phi_i(\mathbf{r})$.

After the impulsive force is applied at $t = 0$, a time evolution calculation is carried out without any external fields,

$i\hbar\frac{\partial}{\partial t}\psi_i(\mathbf{r},t) = H_{\rm KS}(t)\psi_i(\mathbf{r},t)$.

During the time evolution, the electric dipole moment given by

$p_z(t) = \int d\mathbf{r}(-ez)\sum_i |\psi_i(\mathbf{r},t)|^2$,

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 by

$\tilde{\alpha}_{zz}(\omega) = -\frac{e}{I}\int dt e^{i\omega t}p_z(t)$.

Input files

To run the code, following files are necessary:

file name	description
C2H2_response.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
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

First three files are prepared in the directory `SALMON/samples/exercise_02_C2H2_lr/`. The file `C2H2_rt_response.inp` that contains input keywords and their values. The pseudopotential files should be the same as those used in the ground state calculation. In the directory `restart`, those files created in the ground state calculation and stored in the directory `data_for_restart` are included. Therefore, copy the directory as `cp -R data_for_restart restart` if you calculate at the same directory as you did the ground state calculation.

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 input keywords*.

```
!================================================================================
↪===!
! Excercise 02: Polarizability and photoabsorption of C2H2 molecule
↪    !
!--------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained in our manual(see chapter: 'Exercises').
↪    !
```

(continues on next page)

(continued from previous page)

```
!   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 input keywords' in the manual)
↪    !
!-------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':
↪    !
!   Length: 1 [a.u.] = 0.52917721067    [Angstrom]
↪    !
!   Energy: 1 [a.u.] = 27.21138505      [eV]
↪    !
!   Time  : 1 [a.u.] = 0.02418884326505 [fs]
↪    !
!-------------------------------------------------------------------------------
↪---!
! * 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'
/
```

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

```
&units
  !units used in input and output files
```

(continues on next page)

(continued from previous page)

```
  unit_system = 'A_eV_fs'
/
```

unit_system specifies which unit system is used in the input and output files.

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

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

yn_periodic specifies whether or not periodic boundary condition is applied.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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. !
  !------------------------------------------------!
/
```

file_pseudo(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

(continues on next page)

(continued from previous page)

```
  xc = 'PZ'
/
```

xc specifies the exchange-correlation potential to be used in the calculation.

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

  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 64, 64, 64
/
```

dl(i) specifies the spatial grid spacing in i-th direction.
num_rgrid(i) specifies the number of grid points in i-th direction.

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

dt specifies the time step.
nt is the number of time steps for the time propagation.

```
&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 ------------------------------------------------------!
  ! Definition of the incident pulse is written in:                  !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !------------------------------------------------------------------!
/
```

ae_shape1 specifies the envelope of the field. For a linear response calculation, `as_shape1='impulse'` is used. It indicates that a weak impulsive perturbation is applied at $t = 0$.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.

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

de specifies the energy grid size for frequency-domain analysis.
nenergy specifies the number of energy grid points for frequency-domain analysis.

```
&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) !
  !-------------------------------------------------------------!
/
```

&atomic_coor specifies spatial coordinates of atoms.

Execusion

Before execusion, remember to copy the directory `restart` that is created in the ground state calculation as `data_for_restart` in the present directory. In a multiprocess environment, calculation will be executed as:

```
$ mpiexec -n NPROC salmon < C2H2_rt_response.inp > C2H2_rt_response.out
```

where NPROC is the number of MPI processes. A standard output will be stored in the file `C2H2_rt_response.out`.

Output files

After the calculation, following output files are created in the directory that you run the code in addition to the standard output file,

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	total energy and electronic excitation 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_1/exercise_zip_files/02_C2H2_lr.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###################################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                                Version 2.0.1
###################################################################################
  Libxc: [disabled]
   theory= tddft_response

 Total time step      =        5000
 Time step[fs]        =  1.250000000000000E-003
 Energy range         =        3000
 Energy resolution[eV]=  1.000000000000000E-002
 Field strength[a.u.] =  1.000000000000000E-002
   use of real value orbitals =  F
 ======
 .........
```

After that, the time evolution loop starts. At every 10 iteration steps, the time, dipole moments in three Cartesian directions, the total number of electrons, the total energy, and the number of iterations solving the Poisson equation are displayed.

```
 time-step    time[fs]                              Dipole moment(xyz)[A]      electrons␣
↪ Total energy[eV]    iterVh
#-------------------------------------------------------------------------
      10    0.01250000 -0.56521137E-07 -0.28812833E-07 -0.25558983E-01    10.00000000␣
↪    -339.68150366   34
      20    0.02500000 -0.19835467E-06 -0.10147641E-06 -0.45169126E-01     9.99999999␣
↪    -339.68147442   49
      30    0.03750000 -0.37937911E-06 -0.19537418E-06 -0.57843871E-01     9.99999999␣
↪    -339.68146891   45
      40    0.05000000 -0.56465010E-06 -0.29324906E-06 -0.64072126E-01     9.99999999␣
↪    -339.68146804   38
      50    0.06250000 -0.73343753E-06 -0.38431758E-06 -0.65208422E-01     9.99999999␣
↪    -339.68146679   25
```

(continues on next page)

(continued from previous page)

```
      60    0.07500000 -0.87559727E-06 -0.46276791E-06 -0.62464066E-01     9.99999999␣
↪     -339.68146321   35
      70    0.08750000 -0.98769124E-06 -0.52594670E-06 -0.56740338E-01     9.99999998␣
↪     -339.68145535   20
      80    0.10000000 -0.10701350E-05 -0.57309375E-06 -0.48483747E-01     9.99999998␣
↪     -339.68144840   40
      90    0.11250000 -0.11253992E-05 -0.60455485E-06 -0.38296037E-01     9.99999998␣
↪     -339.68144186   21
```

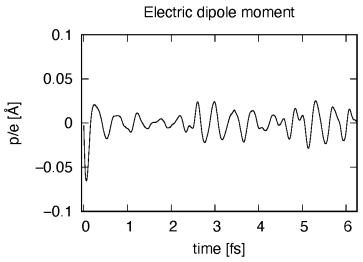
Explanations of other output files are given below:

C2H2_rt.data

Results of time evolution calculation for vector potential, electric field, and dipole moment. In the first several lines, explanations of included data are given.

```
# 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]
```

Using first column (time in femtosecond) and 19th column (dipole moment in z direction), the following graph can be drawn.



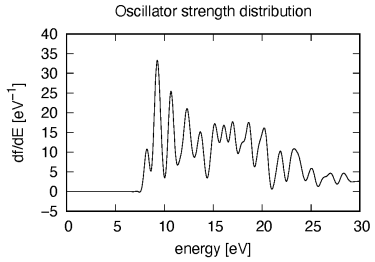
The dipole moment shows oscillations in femtosecond time scale that reflec electronic excitations.

C2H2_response.data

Time-frequency Fourier transformation of the dipole moment gives the polarizability and the strength function.

```
# 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]
```

Using first column (energy in electron-volt) and 10th column (oscillator strength distribution in z direction), the following graph can be drawn.



There appears many peaks above the HOMO-LUMO gap energy. The strong excitation appears at around 9.3 eV.

C2H2_rt_energy.data

Energies are stored as functions of time.

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

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

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 specified by the following vector potential will be used,

$A(t) = -\frac{E_0}{\omega}\hat{z}\cos^2\frac{\pi}{T}\left(t-\frac{T}{2}\right)\sin\omega\left(t-\frac{T}{2}\right), \quad (0<t<T).$

The electric field is given by $E(t) = -(1/c)(dA(t)/dt)$. The parameters that characterize the pulsed field such as the amplitude E_0, frequency ω, pulse duration T, polarization direction $\hat{z}$, are specified in the input file. In the time dependent Kohn-Sham equation, the external field is included as the scalar potential, $V_{\rm ext}(\mathbf{r},t) = eE(t)z$.

Input files

To run the code, following files are necessary:

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*)

First three files are prepared in the directory `SALMON/samples/exercise_03_C2H2_rt/`. The file `C2H2_rt_pulse.inp` that contains input keywords and their values. The pseudopotential files should be the same

as those used in the ground state calculation. In the directory `restart`, those files created in the ground state calculation and stored in the directory `data_for_restart` are included. Therefore, copy the directory as `cp -R data_for_restart restart` if you calculate at the same directory as you did the ground state calculation.

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 input keywords*.

```
!#############################################################################################
↪###!
! Excercise 03:  Electron dynamics in C2H2 molecule under a pulsed electric field          ␣
↪    !
!---------------------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                                  ␣
↪    !
!---------------------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                                      ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                                        ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                              ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                              ␣
↪    !
!---------------------------------------------------------------------------------------------
↪---!
! * 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'
/
```

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

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

yn_periodic specifies whether or not periodic boundary condition is applied.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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. !
  !------------------------------------------------!
/
```

file_pseudo(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

  !number of spatial grids(x,y,z)
  num_rgrid(1:3) = 64, 64, 64
/
```

dl(i) specifies the spatial grid spacing in i-th direction.
num_rgrid(i) specifies the number of grid points in i-th direction.

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

dt specifies the time step.
nt is the number of time steps for the time propagation.

```
&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 5.00d13

  !duration of the incident pulse
  tw1 = 6.00d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 3.10d0
```

(continues on next page)

(continued from previous page)

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

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.

```
&analysis
  !energy grid size and number of energy grids for output files
  de      = 1.0d-2
  nenergy = 10000
/
```

de specifies the energy grid size for frequency-domain analysis.
nenergy specifies the number of energy grid points for frequency-domain analysis.

```
&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) !
  !-----------------------------------------------------------!
/
```

&atomic_coor specifies spatial coordinates of atoms.

Execusion

Before execusion, remember to copy the directory `restart` that is created in the ground state calculation as `data_for_restart` in the present directory. In a multiprocess environment, calculation will be executed as:

```
$ mpiexec -n NPROC salmon < C2H2_rt_pulse.inp > C2H2_rt_pulse.out
```

where NPROC is the number of MPI processes. A standard output will be stored in the file `C2H2_rt_pulse.out`.

Output files

After the calculation, following output files are created in the directory that you run the code in addition to the standard output file,

file name	description
C2H2_pulse.data	time-frequency Fourier transform of dipole moment
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	total energy and electronic excitation 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_1/exercise_zip_files/03_C2H2_rt.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###################################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                               Version 2.0.1
###################################################################################
  Libxc: [disabled]
   theory= tddft_pulse

 Total time step      =         5000
 Time step[fs]        =  1.250000000000000E-003
 Energy range         =        10000
 Energy resolution[eV]=  1.000000000000000E-002
Laser frequency     = 3.10[eV]
Pulse width of laser=      6.00000000[fs]
Laser intensity     =  0.50000000E+14[W/cm^2]
   use of real value orbitals =  F
 ======
 ........
```

After that, the time evolution loop starts. At every 10 iteration steps, the time, dipole moments in three Cartesian directions, the total number of electrons, the total energy, and the number of iterations solving the Poisson equation are displayed.

```
 time-step    time[fs]                                Dipole moment(xyz)[A]      electrons␣
↪ Total energy[eV]    iterVh
#-------------------------------------------------------------------------
      10    0.01250000 -0.57275542E-07 -0.29197105E-07 -0.74600728E-06    10.00000000␣
↪    -339.69524047    1
      20    0.02500000 -0.20616352E-06 -0.10537273E-06 -0.10256205E-04    10.00000000␣
↪    -339.69524348    1
```

(continues on next page)

(continued from previous page)

```
      30    0.03750000 -0.40063325E-06 -0.20597522E-06 -0.47397133E-04    10.00000000␣
↪     -339.69524090    3
      40    0.05000000 -0.59093535E-06 -0.30630513E-06 -0.13774845E-03    10.00000000␣
↪     -339.69524287    1
      50    0.06250000 -0.75588343E-06 -0.39552925E-06 -0.31097825E-03    10.00000000␣
↪     -339.69523949    5
      60    0.07500000 -0.89221538E-06 -0.47142217E-06 -0.59735355E-03    10.00000000␣
↪     -339.69523784   11
      70    0.08750000 -0.99769538E-06 -0.53192187E-06 -0.10253308E-02    10.00000000␣
↪     -339.69523285    5
      80    0.10000000 -0.10738281E-05 -0.57676878E-06 -0.16195168E-02     9.99999999␣
↪     -339.69522482   19
      90    0.11250000 -0.11250289E-05 -0.60722757E-06 -0.23985719E-02     9.99999999␣
↪     -339.69521092    2
```

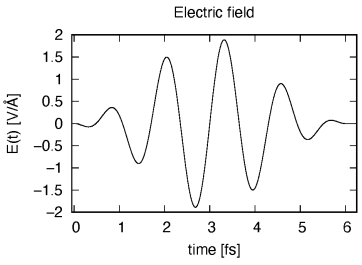
Explanations of other output files are given below:

C2H2_rt.data

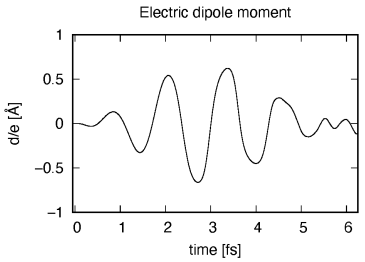
Results of time evolution calculation for vector potential, electric field, and dipole moment. In the first several lines, explanations of data included data are given.

```
# 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]
```

The applied electric field is drawn using the first column (time in femtosecond) and the 7th column (electric field in z direction in Volt per Angstrom).



The induced dipole moment is drawn using the first column (time in femtosecond) and 19th column (dipole moment in z direction). It shows an oscillation similar to the applied electric field. However, the response is not linear since the applied electric field is rather strong.

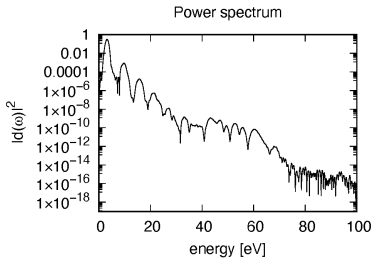


C2H2_pulse.data

Time-frequency Fourier transformation of the dipole moment. In the first several lines, explanations of data included data are given.

```
# 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]
```

The spectrum of the induced dipole moment, $|d(\omega)|^2$ is shown in logarithmic scale as a function of the energy, $\hbar\omega$. High harmonic generations are visible in the spectrum.

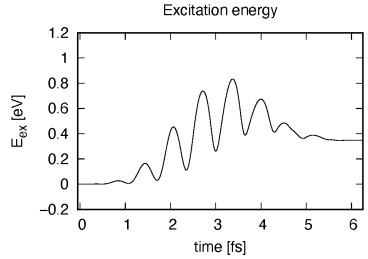


C2H2_rt_energy.data

Energies are stored as functions of time. In the first several lines, explanations of data included data are given.

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

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively. The figure below shows the electronic excitation energy as a function of time, using the first column (time in femtosecond) and the 3rd column (*Eall-Eall0*). Although the frequency is below the HOMO-LUMO gap energy, electronic excitations take place because of nonlinear absorption process.



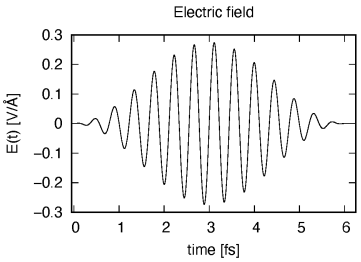
Additional exercise

If we change parameters of the applied electric field, we find a drastic change in the electronic excitations. In the example below, we increase the intensity from `I_wcm2_1 = 5.00d13` to `I_wcm2_1 = 1.00d12` and changes the frequency from `omega1 = 3.10d0` to `omega1 = 9.28d0`. The new frequency corresponds to the resonant excitation energy seen in the linear response analysis shown in in *Exercise-2*.

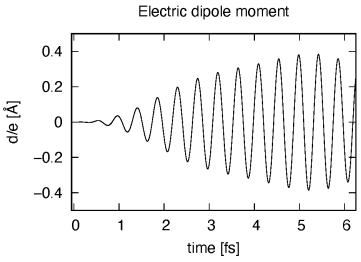
The change in the input file is shown below.

```
&emfield
  !envelope shape of the incident pulse('Ecos2': cos^2 type envelope for scalar␣
↪potential)
  ae_shape1 = 'Acos2'

  !peak intensity(W/cm^2) of the incident pulse
  I_wcm2_1 = 1.00d12

  !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
```

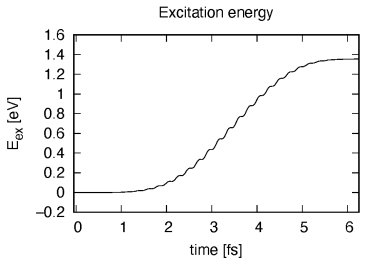
The applied electric field shows a rapid oscillation.



The induced dipole moment also shows a rapid oscillation and does not decrease even though the electric field decreases. This is because the frequency of the applied electric field coincides with the excitation energy of the molecule.



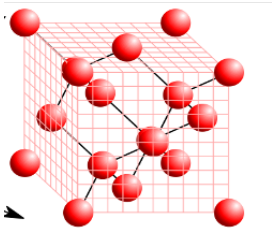
The electronic excitation energy also shows a monotonic increase. Although the strength of the applied electric field is much smaller than the previous case, the amount of the excitation energy is larger, again due to the resonant excitation.



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 that has a diamond structure. A cubic unit cell that contains eight silicon atoms is adopted in the calculation.



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 the directory `SALMON/samples/exercise_04_bulkSi_gs/` 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 input keywords*.

```
!#############################################################################
↪###!
! Excercise 04: Ground state of crystalline silicon(periodic solids)
↪    !
!-----------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                                 ␣
↪    !
!-------------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                                     ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                                       ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                             ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                             ␣
↪    !
!#####################################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

  !grid box size(x,y,z)
```

(continues on next page)

(continued from previous page)

```
  al(1:3) = 5.43d0, 5.43d0, 5.43d0

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

yn_periodic specifies whether or not periodic boundary condition is applied.
al(i) specifies the side length of the unit cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

num_rgrid(i) specifies the number of real-space grid point in i-th direction.

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

num_kgrid(i) specifies the number of k-points for i-th direction discretizing the Brillouin zone.

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

nscf specifies the maximum number of SCF iterations.
threshold specifies the threshold to judge the convergence.

```
&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) !
  !-------------------------------------------------------------!
/
```

&atomic_red_coor specifies spatial coordinates of atoms in reduced coordinate system.

Execusion

In a multiprocess environment, calculation will be executed as:

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

where NPROC is the number of MPI processes. A standard output will be stored in the file `Si_gs.out`.

Output files

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

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_1/exercise_zip_files/04_bulkSi_gs.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###############################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                               Version 2.0.1
###############################################################################
  Libxc: [disabled]
   theory= dft
   use of real value orbitals =  F
 r-space parallelization: off
 ======
 MPI distribution:
   nproc_k     :          16
   nproc_ob    :           1
   nproc_rgrid :           1           1           1
 OpenMP parallelization:
   number of threads :          64
 .........
```

After that, the SCF loop starts. At each iteration step, the total energy as well as orbital energies and some other quantities are displayed.

```
-----------------------------------------------
 iter=     1     Total Energy=       314.78493406     Gap=   -95.88543131
 k=           1
     1        37.5762      2        63.8589      3        58.1850      4        43.
↪0042
     5        61.5347      6        29.5604      7        41.5986      8        39.
↪3545
     9        48.5641     10        68.0003     11        75.5196     12        85.
↪4113
 ..........
    21        94.1224     22        53.0821     23        72.0170     24        46.
↪7797
    25        88.6077     26        98.2698     27        42.8071     28        65.
↪0812
    29        60.3648     30        39.6787     31        83.5629     32        62.
↪7365
```

(continues on next page)

(continued from previous page)

```
 iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec        =      1 0.49478519E+00
 Ne=   32.0000000000000
 -----------------------------------------------
 iter=     2     Total Energy=        62.72724688     Gap=   -77.31200657
 k=          1
     1        14.4913      2        32.6869      3        30.3561      4        20.
↪6816
     5        30.3907      6        16.9184      7        22.2967      8        18.
↪5338
     9        29.0117     10        41.9687     11        42.3490     12        54.
↪6262
..........
```

When the convergence criterion is satisfied, the SCF calculation ends.

```
 iter=    60     Total Energy=      -850.76385275     Gap=     1.06020364
 k=          1
     1        -3.7745      2        -3.0158      3        -3.0158      4        -3.
↪0158
     5        -0.4300      6        -0.4300      7        -0.4300      8         0.
↪3765
     9         3.9530     10         3.9530     11         3.9530     12         4.
↪6110
..........
    21         9.6233     22         9.6233     23         9.6956     24         9.
↪9111
    25        11.0259     26        11.0259     27        11.4165     28        11.
↪5976
    29        11.9826     30        11.9887     31        12.0967     32        12.
↪3585

 iter and int_x|rho_i(x)-rho_i-1(x)|dx/nelec        =     60 0.77889300E-09
 Ne=   32.0000000000000
  #GS converged at    61  : 0.77889300E-09
 ===== force =====
     1  0.60775985E-08  0.15425240E-07 -0.22474791E-07
     2 -0.10689345E-06  0.88233132E-07  0.35122981E-09
     3  0.39762202E-07 -0.23921918E-07  0.11855231E-07
     4 -0.79441825E-07 -0.28978042E-07 -0.34109698E-07
     5  0.37990526E-07  0.67211638E-08  0.20384753E-07
     6  0.96418986E-07 -0.70404285E-07  0.10198912E-06
     7  0.16145540E-07  0.30561301E-07 -0.63738382E-07
     8  0.26042178E-07  0.30977639E-07 -0.40587816E-07
 band information-----------------------------------------
 Bottom of VB -0.194818046940532
 Top of VB  0.216611832367042
 Bottom of CB  0.255573599266334
 Top of CB  0.533770712688357
 Fundamental gap  3.896176689929157E-002
 BG between same k-point  3.896176691206812E-002
 Physicaly upper bound of CB for DOS  0.453918744010958
 Physicaly upper bound of eps(omega)  0.609598295602846
 -------------------------------------------------------
 Bottom of VB[eV]  -5.30126888998779
 Top of VB[eV]   5.89430797692564
 Bottom of CB[eV]   6.95451161825061
```

(continues on next page)

(continued from previous page)

```
 Top of CB[eV]   14.5246403913758
 Fundamental gap[eV]   1.06020364132497
 BG between same k-point[eV]   1.06020364167264
 Physicaly upper bound of CB for DOS[eV]   12.3517577246945
 Physicaly upper bound of eps(omega)[eV]   16.5880139474728
 -------------------------------------------------------
  writing restart data...
  writing completed.
```

In the directory `data_for_restart`, files that will be used in the next-step time evolution calculations are stored.

Other output files include following information.

Si_info.data

Orbital and total energies as well as parameters specified in the input file.

```
Total number of iteration =           60

Number of states =           32
Number of electrons =           32

Total energy (eV) =   -850.763852754463
1-particle energies (eV)
    1        -3.7745      2        -3.0158      3        -3.0158      4        -3.0158
    5        -0.4300      6        -0.4300      7        -0.4300      8         0.3765
    9         3.9530     10         3.9530     11         3.9530     12         4.6110
```

Si_eigen.data

Orbital energies.

```
#esp: single-particle energies (eigen energies)
#occ: occupation numbers, io: orbital index
# 1:io, 2:esp[eV], 3:occ
k=     1,  spin=     1
     1  -0.3774501171245852E+001   0.2000000000000000E+001
     2  -0.3015778973884847E+001   0.2000000000000000E+001
     3  -0.3015778969794385E+001   0.2000000000000000E+001
```

Si_k.data

Data of k-points.

```
# 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]
     1 -0.375000000000000E+000 -0.375000000000000E+000 -0.375000000000000E+000  0.
↪156250000000000E-001
     2 -0.125000000000000E+000 -0.375000000000000E+000 -0.375000000000000E+000  0.
↪156250000000000E-001
     3  0.125000000000000E+000 -0.375000000000000E+000 -0.375000000000000E+000  0.
↪156250000000000E-001
```

3.3.2 Exercise-5: Dielectric function of crystalline silicon

In this exercise, we learn the linear response calculation of the crystalline silicon. A cubic unit cell that contains eight silicon atoms is used in the calculation. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*.

In this exercise, we calculate a dielectric function of silicon as a final object. We first summarize definitions of relevant quantities. We introduce a conductivity in time domain, $\sigma_{\mu\nu}(t)$, where μ, ν indicate Cartesian components, $\mu, \nu = x, y, z$. It relates the applied electric field $E_\nu(t)$ with the induced current density averaged over the unit cell volume, $J_\mu(t)$,

$J_\mu(t) = \sum_{\nu=x,y,z} \int dt' \sigma_{\mu\nu}(t-t') E_\nu(t')$.

Integrating the current density over time, we obtain the polarization density as a functioon of time,

$P_\mu(t) = \int^t dt' J_\mu(t')$.

Then, the dielectric function is introduced by

$D_\mu(t) = E_\mu(t) + 4\pi P_\mu(t) = \sum_\nu \int^t dt' \epsilon_{\mu\nu}(t-t') E_\nu(t')$.

Frequency-dependent dielectric function $\epsilon_{\mu\nu}(\omega)$ is obtained from $\epsilon_{\mu\nu}(t)$ by taking time-frequency Fourier transformation.

In SALMON, the dielectric function is calculated in the following way. First the ground state Bloch orbitals $u_{n\mathbf{k}}(\mathbf{r})$ that satisfies the Kohn-Sham equation,

$H_{\mathbf{k}} u_{n\mathbf{k}}(\mathbf{r}) = \epsilon_{n\mathbf{k}}(\mathbf{r})$,

is calculated. Then an impulsive force characterized by the magnitude of the impulse I is applied to all electrons in z direction. This is equivalent to shift the wave vector by $\mathbf{k} \to \mathbf{k} + I/\hbar\hat{z}$, where $\hat{z}$ is a unit vector in z direction. We make a time evolution calculation with the shifted wave vector as

$i\hbar \frac{\partial}{\partial t} u_{n\mathbf{k}}(\mathbf{r}, t) = H_{\mathbf{k}+I/\hbar\hat{z}}(t) u_{n\mathbf{k}}(\mathbf{r}, t)$.

During the time evolution, the electric current density given by

$\mathbf{J}(t) = \frac{-e}{m\Omega} \int d\mathbf{r} u^*_{n\mathbf{k}} \left(-i\hbar\nabla + \hbar\mathbf{k} + I\hat{z}\right) u_{n\mathbf{k}} + \delta\mathbf{J}(t)$.

is monitored, where Ω is the volume of the unit cell and $\delta\mathbf{J}(t)$ is a current component coming from nonlocal pseudopootential.

After the time evolution calculation, a time-frequency Fourier transformation is carried out for the electric current density to obtain the frequency-dependent conductivity by

$\tilde{\sigma}_{zz}(\omega) = -\frac{e}{I} \int dt e^{i\omega t} J_z(t)$.

The dielectric function and the conductivity is related in frequency representation by

$\epsilon_{\mu\nu}(\omega) = \delta_{\mu\nu} + \frac{4\pi i \sigma_{\mu\nu}(\omega)}{\omega}$.

We note that the dielectric function of a crystalline silicon is isotropic, $\epsilon_{\mu\nu} = \delta_{\mu\nu}\epsilon(\omega)$.

Input files

To run the code, following files are necessary:

file name	description
C2H2_response.inp	input file that contains input keywords and their values
Si_rps.dat	pseodupotential file for silicon atom
restart	directory created in the ground state calculation (rename the directory from *data_for_restart* to *restart*)

First two files are prepared in the directory `SALMON/samples/exercise_05_bulkSi_lr/`. The file `Si_rt_response.inp` contains input keywords and their values. The pseudoopotential file should be the same as that used in the ground state calculation. In the directory `restart`, those files created in the ground state calculation and stored in the directory `data_for_restart` are included. Therefore, coopy the directory as `cp -R data_for_restart restart` if you calculate at the same directory as you did the ground state calculation.

In the input file `Si_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 input keywords*.

```
!#################################################################################
↪###!
! Excercise 05: Dielectric function of crystalline silicon                      ␣
↪    !
!--------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                       ␣
↪    !
!--------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                          ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                             ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                   ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                   ␣
↪    !
!--------------------------------------------------------------------------------
↪---!
! * 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.                                                  ␣
↪    !
```

(continues on next page)

(continued from previous page)

```
!#########################################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

  !grid box size(x,y,z)
  al(1:3) = 5.43d0, 5.43d0, 5.43d0

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

yn_periodic specifies whether or not periodic boundary condition is applied.
al(i) specifies the side length of the unit cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

num_rgrid(i) specifies the number of real-space grid point in i-th direction.

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

num_kgrid(i) specifies the number of k-points for i-th direction discretizing the Brillouin zone.

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.002d0
  nt = 6000
/
```

dt specifies the time step.
nt is the number of time steps for the time propagation.

```
&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 ------------------------------------------------------!
  ! Definition of the incident pulse is written in:                  !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

ae_shape1 specifies the envelope of the field. For a linear response calculation, `as_shape1='impulse'` is used. It indicates that a weak impulsive perturbation is applied at $t = 0$.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.

```
&analysis
  !energy grid size and number of energy grids for output files
  de      = 0.01d0
  nenergy = 2000
/
```

de specifies the energy grid size for frequency-domain analysis.
nenergy specifies the number of energy grid points for frequency-domain analysis.

```
&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) !
  !-------------------------------------------------------------!
/
```

&atomic_red_coor specifies spatial coordinates of atoms in reduced coordinate system.

Execusion

In a multiprocess environment, calculation will be executed as:

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

where NPROC is the number of MPI processes. A standard output will be stored in the file `Si_rt_response.out`.

Output files

After the calculation, following output files are created in the directory that you run the code in addition to the standard output file,

file name	description
Si_response.data	conductivity and dielectric function as functions of energy
Si_rt.data	vector potential, electric field, and matter current as functions of time
Si_rt_energy	total energy and electronic excitation 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_1/exercise_zip_files/05_bulkSi_lr.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###############################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                               Version 2.0.1
###############################################################################
  Libxc: [disabled]
   theory= tddft_response

 Total time step      =         6000
 Time step[fs]        =  2.000000000000000E-003
 Energy range         =         2000
 Energy resolution[eV]=  1.000000000000000E-002
 Field strength[a.u.] =  1.000000000000000E-002
   use of real value orbitals =  F
 r-space parallelization: off
 ======
 ........
```

After that, the time evolution loop starts. At every 10 iteration steps, electric current density in three Cartesian direction, the total number of electrons, and total energy are displayed.

```
  time-step  time[fs]                                Current(xyz)[a.u.]      electrons␣
↪Total energy[eV]
#-------------------------------------------------------------------------
      10    0.02000000  0.11911770E-11 -0.40018285E-13  0.24902126E-03    32.00000000␣
↪    -850.72273308
      20    0.04000000  0.17745321E-11  0.13712105E-12  0.21977876E-03    31.99999999␣
↪    -850.72273319
```

(continues on next page)

(continued from previous page)

```
      30    0.06000000  0.31016197E-11  0.24481043E-12  0.20049151E-03    31.99999999␣
↪     -850.72272966
      40    0.08000000  0.36611565E-11  0.49184860E-12  0.17937042E-03    31.99999999␣
↪     -850.72272925
      50    0.10000000  0.36920991E-11  0.63805259E-12  0.15246564E-03    31.99999998␣
↪     -850.72272922
      60    0.12000000  0.32347636E-11  0.11280947E-11  0.12248647E-03    31.99999998␣
↪     -850.72272655
      70    0.14000000  0.25978450E-11  0.15550074E-11  0.91933957E-04    31.99999998␣
↪     -850.72272293
      80    0.16000000  0.20087959E-11  0.17983589E-11  0.62968342E-04    31.99999997␣
↪     -850.72272036
      90    0.18000000  0.90623268E-12  0.18067974E-11  0.38824129E-04    31.99999997␣
↪     -850.72271918
```

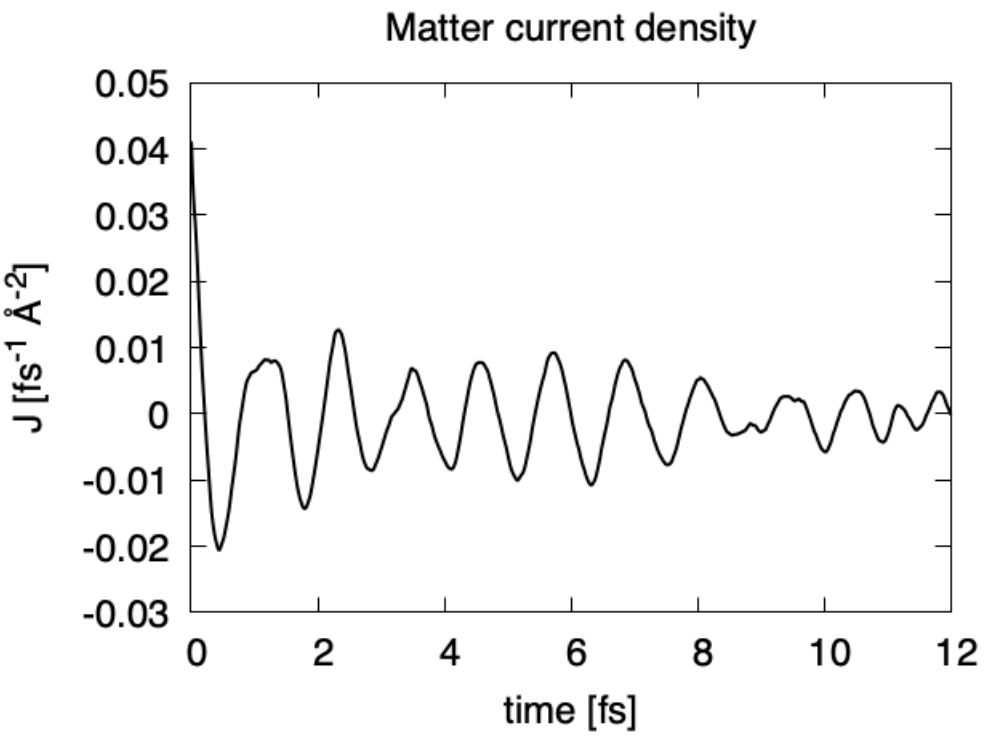
Explanations of other output files are given below:

Si_rt.data

Results of time evolution calculation for vector potential, electric field, and matter current density are shown. In the first several lines, explanations of included data are given.

```
# 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[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:Jm_x[1/fs*Angstrom^2]
# 15:Jm_y[1/fs*Angstrom^2] 16:Jm_z[1/fs*Angstrom^2]
```

Using first column (time in femtosecond) and 16th column (matter current density in z direction), the following graph can be drawn.

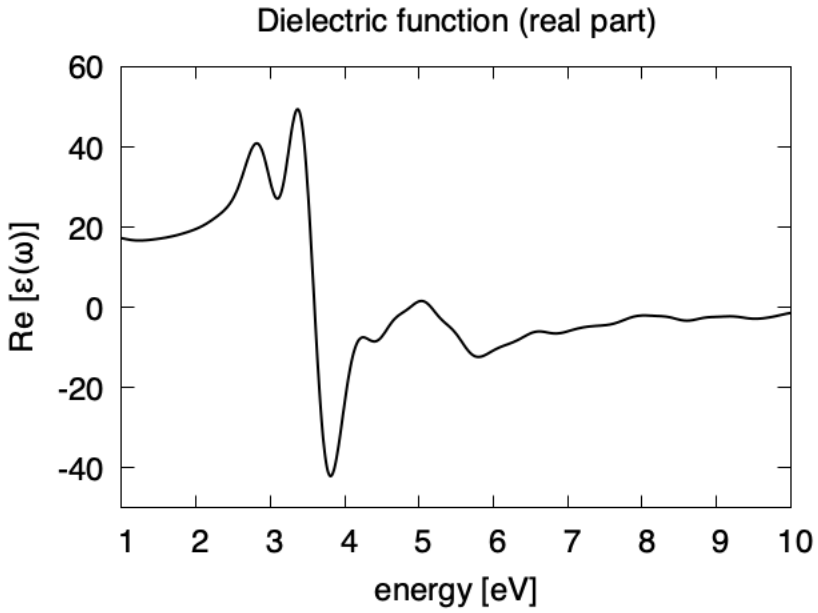


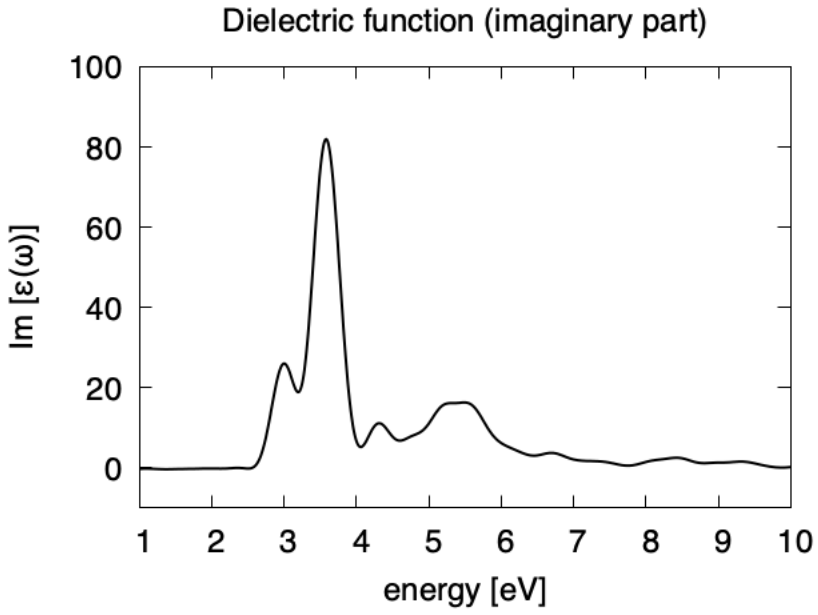
Si_response.data

Time-frequency Fourier transformation of the macroscopic current density gives the conductivity of the system. The dielectric function is then calculated from the conductivity. They are stored in this file.

```
# Fourier-transform spectra:
# sigma: Conductivity
# eps: Dielectric constant
# 1:Energy[eV] 2:Re(sigma_x)[1/fs*V*Angstrom] 3:Re(sigma_y)[1/fs*V*Angstrom]
# 4:Re(sigma_z)[1/fs*V*Angstrom] 5:Im(sigma_x)[1/fs*V*Angstrom]
# 6:Im(sigma_y)[1/fs*V*Angstrom] 7:Im(sigma_z)[1/fs*V*Angstrom] 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]
```

Using first column (energy in eV) and 10th (real part of the dielectric function) and 13th (imaginary part), we obtain the following graph.





The imaginary part appears above the direct bandgap energy that is about 2.4 eV in the present calculation using local density approximation. Dielectric function below 1 eV are not accurate and and are not shown.

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[fs] 2:Eall[eV] 3:Eall-Eall0[eV]
```

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 crystalline silicon. A cubic unit cell that contains eight silicon atoms is used in the calculation. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*.

In the calculation, a pulsed electric field specified by the following vector potential will be used,

$A(t) = -\frac{E_0}{\omega}\hat{z}\cos^2\frac{\pi}{T}\left(t-\frac{T}{2}\right)\sin\omega\left(t-\frac{T}{2}\right), \quad (0<t<T).$

The electric field is given by $E(t) = -(1/c)(dA(t)/dt)$. The parameters that characterize the pulsed field such as the amplitude E_0, frequency ω, pulse duration T, polarization direction $\hat{z}$, are specified in the input file. Time-dependent Kohn-Sham equation for Bloch orbitals are calculated in real time,

$i\hbar\frac{\partial}{\partial t}u_{n\mathbf{k}}(\mathbf{r},t) = H_{\mathbf{k}+(e/\hbar c)\mathbf{A}(t)}u_{n\mathbf{k}}(\mathbf{r},t).$

Input files

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

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*)

First two files are prepared in the directory `SALMON/samples/exercise_06_bulkSi_rt/`. The file `Si_rt_pulse.inp` contains input keywords and their values. The pseudoopotential file should be the same as that used in the ground state calculation. In the directory `restart`, those files created in the ground state calculation and stored in the directory `data_for_restart` are included. Therefore, coopy the directory as `cp -R data_for_restart restart` if you calculate at the same directory as you did the ground state calculation.

In the input file `Si_rt_pulse.inp`, input keywords are specified. Most of them are mandatory to execute the electron dynamics 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 input keywords*.

```
!################################################################################
↪###!
! Excercise 06: Electron dynamics in crystalline silicon under a pulsed electric␣
↪field   !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                             ␣
↪    !
!-------------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                                 ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                                    ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                          ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                          ␣
↪    !
!-------------------------------------------------------------------------------------
↪---!
! * 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'
/
```

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

  !grid box size(x,y,z)
  al(1:3) = 5.43d0, 5.43d0, 5.43d0

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

yn_periodic specifies whether or not periodic boundary condition is applied.
al(i) specifies the side length of the unit cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

num_rgrid(i) specifies the number of real-space grid point in i-th direction.

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

num_kgrid(i) specifies the number of k-points for i-th direction discretizing the Brillouin zone.

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.002d0
  nt = 6000
/
```

dt specifies the time step.
nt is the number of time steps for the time propagation.

```
&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 = 10.672d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 1.55d0

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

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.

```
&analysis
  !energy grid size and number of energy grids for output files
  de      = 0.01d0
  nenergy = 3000
/
```

de specifies the energy grid size for frequency-domain analysis.
nenergy specifies the number of energy grid points for frequency-domain analysis.

```
&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) !
  !-----------------------------------------------------------------!
/
```

&atomic_red_coor specifies spatial coordinates of atoms in reduced coordinate system.

Execusion

In a multiprocess environment, calculation will be executed as:

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

where NPROC is the number of MPI processes. A standard output will be stored in the file `Si_rt_pulse.out`.

Output files

After the calculation, following output files are created in the directory that you run the code in addition to the standard output file,

file name	description
Si_pulse.data	time-frequency Fourier transform of matter current and electric field
Si_rt.data	vector potential, electric field, and matter current as functions of time
Si_rt_energy	total energy and electronic excitation 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_1/exercise_zip_files/06_bulkSi_rt.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###################################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                               Version 2.0.1
###################################################################################
  Libxc: [disabled]
   theory= tddft_pulse

 Total time step      =         6000
 Time step[fs]        =  2.000000000000000E-003
 Energy range         =         3000
 Energy resolution[eV]=  1.000000000000000E-002
Laser frequency     = 1.55[eV]
Pulse width of laser=     10.67200000[fs]
Laser intensity     =  0.10000000E+13[W/cm^2]
   use of real value orbitals =  F
 r-space parallelization: off
 ======
 ........
```

After that, the time evolution loop starts. At every 10 iterations, the time, current in three Cartesian directions, the number of electrons, and the total energy are displayed.

```
  time-step  time[fs]                                  Current(xyz)[a.u.]      electrons␣
↪Total energy[eV]
#-------------------------------------------------------------------------
      10    0.02000000  0.11847131E-11 -0.47534543E-13 -0.43120486E-08    32.00000000␣
↪    -850.76385276
      20    0.04000000  0.17733186E-11  0.12820952E-12 -0.33012195E-07    32.00000000␣
↪    -850.76385276
      30    0.06000000  0.30965601E-11  0.23626542E-12 -0.10736819E-06    32.00000000␣
↪    -850.76385275
      40    0.08000000  0.36612711E-11  0.47687574E-12 -0.24607217E-06    32.00000000␣
↪    -850.76385272
      50    0.10000000  0.36958981E-11  0.62315158E-12 -0.46548014E-06    32.00000000␣
↪    -850.76385263
      60    0.12000000  0.32186097E-11  0.11429104E-11 -0.77911390E-06    32.00000000␣
↪    -850.76385239
      70    0.14000000  0.25712602E-11  0.15689467E-11 -0.11971541E-05    32.00000000␣
↪    -850.76385186
      80    0.16000000  0.19447699E-11  0.18250920E-11 -0.17261976E-05    32.00000000␣
↪    -850.76385082
```

(continues on next page)

(continued from previous page)

```
      90    0.18000000  0.80514520E-12  0.18683828E-11 -0.23692381E-05    32.00000000␣
↪     -850.76384896
```

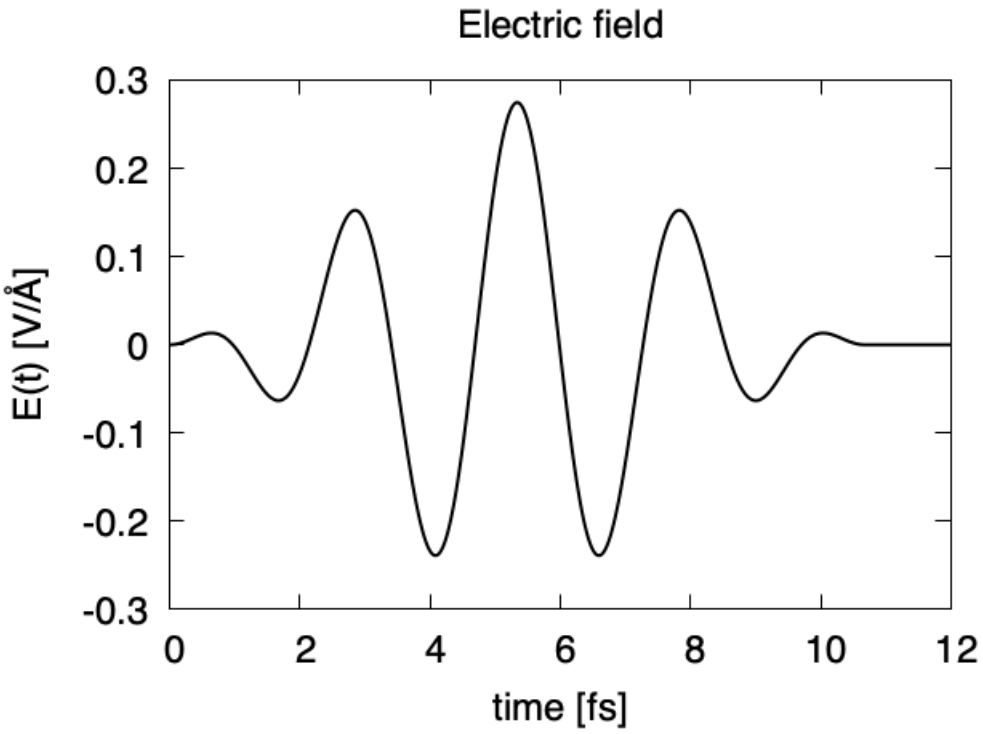
Explanations of other output files are given below:

Si_rt.data

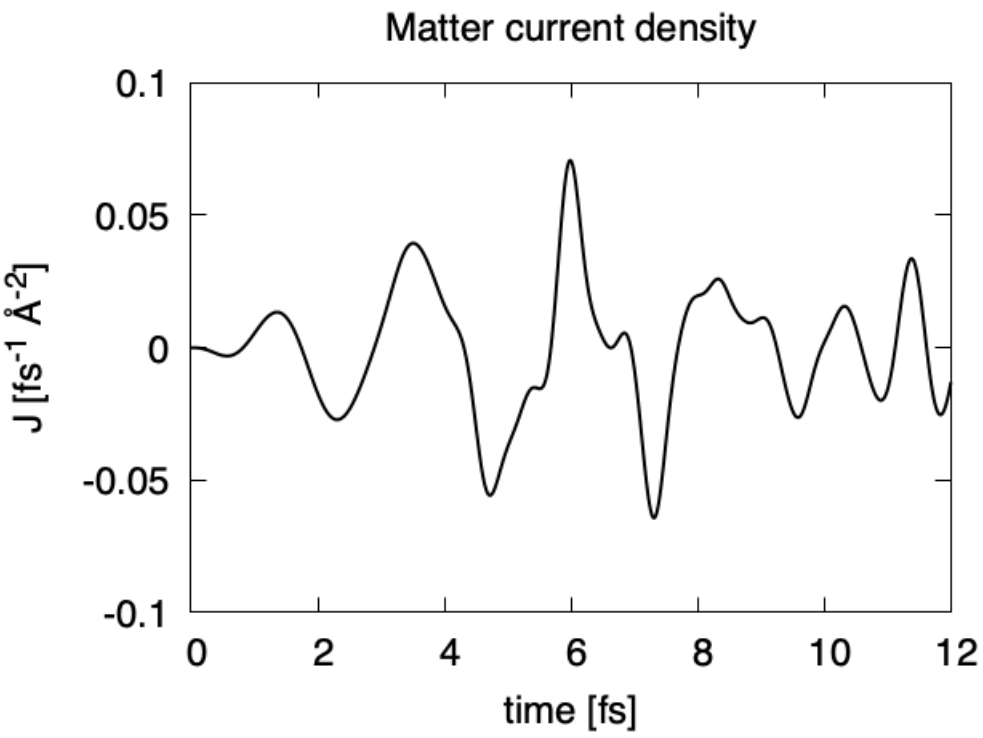
Results of time evolution calculation for vector potential, electric field, and matter current density.

```
# 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[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:Jm_x[1/fs*Angstrom^2] 15:Jm_y[1/fs*Angstrom^2] 16:Jm_z[1/fs*Angstrom^2]
```

The applied electric field is drawn using the first column (time in femtosecond) and the 7th column (electric field in z direction).



The matter current density is drawn using the first column (time in femtosecond) and 16th column (matter current density in z direction).

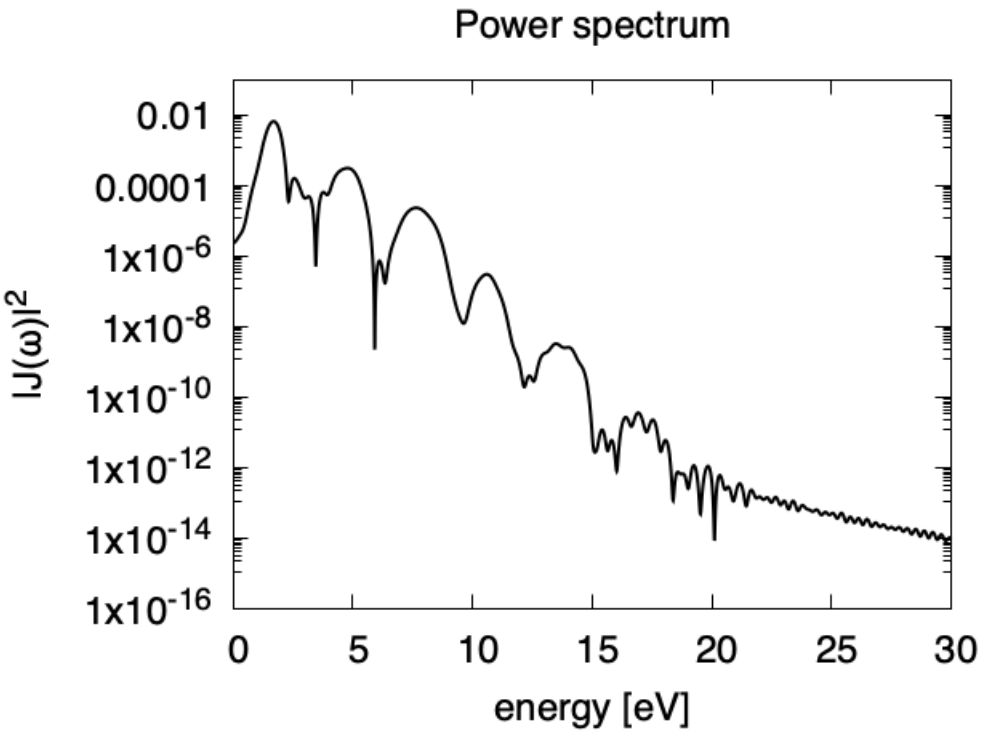


Si_pulse.data

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[eV] 2:Re(Jm_x)[1/Angstrom^2] 3:Re(Jm_y)[1/Angstrom^2] 4:Re(Jm_z)[1/
↪Angstrom^2]
# 5:Im(Jm_x)[1/Angstrom^2] 6:Im(Jm_y)[1/Angstrom^2] 7:Im(Jm_z)[1/Angstrom^2]
# 8:|Jm_x|^2[1/Angstrom^4] 9:|Jm_y|^2[1/Angstrom^4] 10:|Jm_z|^2[1/Angstrom^4]
# 11:Re(E_ext_x)[fs*V/Angstrom] 12:Re(E_ext_y)[fs*V/Angstrom]
# 13:Re(E_ext_z)[fs*V/Angstrom] 14:Im(E_ext_x)[fs*V/Angstrom]
# 15:Im(E_ext_y)[fs*V/Angstrom] 16:Im(E_ext_z)[fs*V/Angstrom]
# 17:|E_ext_x|^2[fs^2*V^2/Angstrom^2] 18:|E_ext_y|^2[fs^2*V^2/Angstrom^2]
# 19:|E_ext_z|^2[fs^2*V^2/Angstrom^2] 20:Re(E_tot_x)[fs*V/Angstrom]
# 21:Re(E_tot_y)[fs*V/Angstrom] 22:Re(E_tot_z)[fs*V/Angstrom]
# 23:Im(E_tot_x)[fs*V/Angstrom] 24:Im(E_tot_y)[fs*V/Angstrom]
# 25:Im(E_tot_z)[fs*V/Angstrom] 26:|E_tot_x|^2[fs^2*V^2/Angstrom^2]
# 27:|E_tot_y|^2[fs^2*V^2/Angstrom^2] 28:|E_tot_z|^2[fs^2*V^2/Angstrom^2]
```

The power spectrum of the matter current density, $|J(\omega)|^2$ is shown in logarithmic scale as a function of the energy, $\hbar\omega$. High harmonic generations are visible in the spectrum.

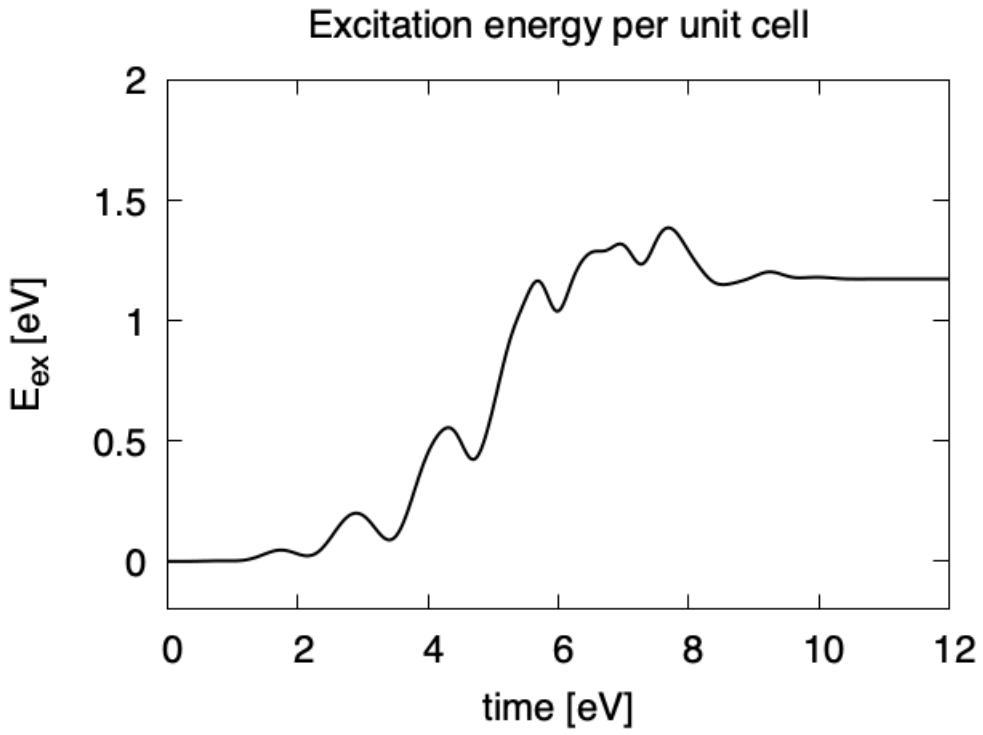


Si_rt_energy

Energies are stored as functions of time.

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

Eall and *Eall-Eall0* are total energy and electronic excitation energy, respectively. The figure below shows the electronic excitation energy per unit cell volume as a function of time, using the first column (time in femtosecond) and the 3rd column (*Eall-Eall0*). Although the frequency is below the direct bandgap of silicon (2.4 eV in the LDA calculation), electronic excitations take place because of nonlinear absorption process.



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 a propagation of pulsed light through a thin film of crystalline silicon. We consider an irradiation of a few-cycle, linearly polarized pulsed light normally on a thin film of 40 nm thickness. This exercise should be carried out after finishing the ground state calculation that was explained in *Exercise-4*.

In the calculation, macroscopic Maxwell equation that describes the light propagation and microscopic time-dependent Kohn-Sham equation that describes the electron dynamics are solved simultaneously. The light propagation is described by a one-dimensional light-propagation equation for the vector potential,

$\frac{1}{c^2}\frac{\partial^2}{\partial t^2}A(X,t) - \frac{\partial^2}{\partial X^2}A(X,t) = \frac{4\pi}{c}I(X,t).$

The direction of the propagation is set to x direction and the polarization of the pulse is set to z direction. The time profile of an incident pulse is given by

$A(t) = -\frac{E_0}{\omega}\hat{z}\cos^2\frac{\pi}{T}\left(t-\frac{T}{2}\right)\sin\omega\left(t-\frac{T}{2}\right), \qquad (0<t<T),$

and is set in the vacuum region in front of the thin film. The parameters that characterize the pulsed field such as the amplitude E_0, frequency ω, pulse duration T are specified in the input file.

To discribe the light propagation, macroscopic coordinate X is discretized as X_i. At each grid point inside the silicon thin film, for which we take eight points $i = 1 \cdots 8$ in this exercise, time-dependent Kohn-Sham equation for Bloch orbitals are calculated in real time,

$i\hbar\frac{\partial}{\partial t}u_{in\mathbf{k}}(\mathbf{r},t) = H_{\mathbf{k}+(e/\hbar c)\mathbf{A}_i(t)}u_{in\mathbf{k}}(\mathbf{r},t).$

From the Bloch orbital $u_{in\mathbf{k}}(\mathbf{r},t)$, we calculate the electric current $I(X_i,t)$. We thus obtain a closed set of equations. Solving these equations simultaneously, we can describe macroscopic light propagation and microscopic electron dynamics at the same time.

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*)

First two files are prepared in the directory `SALMON/samples/exercise_07_bulkSi_multiscale/`. The file `Si_rt_multiscale.inp` contains input keywords and their values. The pseudoopotential file should be the same as that used in the ground state calculation. In the directory `restart`, those files created in the ground state calculation and stored in the directory *data_for_restart* are included. Therefore, coopy the directory as `cp -R data_for_restart restart` if you calculate at the same directory as you did the ground state calculation.

In the input file `Si_rt_multiscale.inp`, input keywords are specified. Most of them are mandatory to execute the electron dynamics 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 input keywords*.

```
!#############################################################################
↪###!
! Excercise 07: Maxwell+TDDFT multiscale simulation                          ␣
↪    !
!               (Pulsed-light propagation through a silicon thin film)       ␣
↪    !
!-----------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                    ␣
↪    !
!-----------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                       ␣
↪    !
!   Length: 1 [a.u.] = 0.52917721067    [Angstrom]                           ␣
↪    !
!   Energy: 1 [a.u.] = 27.21138505      [eV]                                 ␣
↪    !
!   Time  : 1 [a.u.] = 0.02418884326505 [fs]                                 ␣
↪    !
```

(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_04_bulkSi_gs/' and rename the directory to
↪'restart/'!
!   in the current directory.                                                   ␣
↪   !
!###############################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

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

  !grid box size(x,y,z)
  al(1:3) = 5.43d0, 5.43d0, 5.43d0

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

yn_periodic specifies whether or not periodic boundary condition is applied.
al(i) specifies the side length of the unit cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

num_rgrid(i) specifies the number of real-space grid point in i-th direction.

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

num_kgrid(i) specifies the number of k-points for i-th direction discretizing the Brillouin zone.

```
&tgrid
  !time step size and number of time grids(steps)
  dt = 0.002d0
  nt = 8000
/
```

dt specifies the time step.
nt is the number of time steps for the time propagation.

```
&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 = 10.672d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 1.55d0

  !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 written in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.

```
&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 = 50.0d0
```

(continues on next page)

(continued from previous page)

```
  hy_m = 50.0d0
  hz_m = 50.0d0

  !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
/
```

nx_m, *ny_m*, *nz_m* specify the number of macroscopic grid points inside the material.
hx_m, *hy_m*, *hz_m* specify the grid spacing of macroscopic coordinates.
nxvacl_m / *nxvacr_m* specifies the number of grid points in the vacuum region in the left / right side of the material.

```
&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(i,n) specifies the boundary condition for the electromagnetic analysis. The first index i corresponds to the x,y, and z direction. The second index n specifies left or right side of the material.

```
&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) !
  !-------------------------------------------------------------!
/
```

&atomic_red_coor specifies spatial coordinates of atoms in reduced coordinate system.

Execusion

In a multiprocess environment, calculation will be executed as:

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

where NPROC is the number of MPI processes. A standard output will be stored in the file `Si_rt_multiscale.out`.

Output files

After the calculation, following output files and directories are created in the directory that you run the code in addition to the standard output file.

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	total energy and electronic excitation energy at macroscopic position *xxxxxx* as functions of time
Si_m/mxxxxxx/PS_Si_KY_n.dat	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 spatial position
Si_wave.data	waveform of incident, reflected, and transmitted waves

You may download the above files (zipped file) from:
https://salmon-tddft.jp/webmanual/v_2_0_1/exercise_zip_files/07_bulkSi_ms.zip

We first explain the standard output file. In the beginning of the file, input variables used in the calculation are shown.

```
###############################################################################
# SALMON: Scalable Ab-initio Light-Matter simulator for Optics and Nanoscience
#
#                              Version 2.0.1
###############################################################################
  Libxc: [disabled]
   theory= multi_scale_maxwell_tddft
Initializing macropoint:     1-     8

 Total time step      =         8000
 Time step[fs]        =  2.000000000000000E-003
 Energy range         =         1000
```

(continues on next page)

(continued from previous page)

```
 Energy resolution[eV]=  1.000000000000000E-002
Laser frequency     = 1.55[eV]
Pulse width of laser=     10.67200000[fs]
Laser intensity     =  0.10000000E+13[W/cm^2]
   use of real value orbitals =  F
 r-space parallelization: off
 ======
 .........
```

After that, the time evolution loop starts. At every 100 iterations, the step, grid point index, time, current in three Cartesian directions, the number of electrons, and the total energy are displayed.

```
   Step  Macro      Time                              Current       Electrons  Eabs/cell
                      fs                    1/fs*Angstrom^2                         eV
#-------------------------------------------------------------------------------------
↪-----
    100      1     0.200  5.45E-010 -4.60E-011  2.70E-004    32.00000000  2.36E-006
    100      2     0.200  5.45E-010 -1.56E-011  1.83E-004    32.00000000  1.06E-006
    100      3     0.200  5.45E-010  7.19E-012  1.23E-004    32.00000000  4.62E-007
    100      4     0.200  5.45E-010  2.11E-011  8.14E-005    32.00000000  1.97E-007
    100      5     0.200  5.45E-010  2.11E-011  5.28E-005    32.00000000  8.04E-008
    100      6     0.200  5.45E-010  7.20E-012  3.34E-005    32.00000000  3.11E-008
    100      7     0.200  5.45E-010 -1.56E-011  2.03E-005    32.00000000  1.10E-008
    100      8     0.200  5.45E-010 -4.60E-011  1.13E-005    32.00000000  3.27E-009
    200      1     0.400  1.77E-011 -2.93E-013  9.70E-004    32.00000000  5.80E-005
    200      2     0.400  1.78E-011 -3.64E-011  7.50E-004    32.00000000  3.25E-005
    200      3     0.400  1.78E-011 -5.58E-011  5.75E-004    32.00000000  1.80E-005
    200      4     0.400  1.78E-011 -6.66E-011  4.38E-004    32.00000000  9.89E-006
```

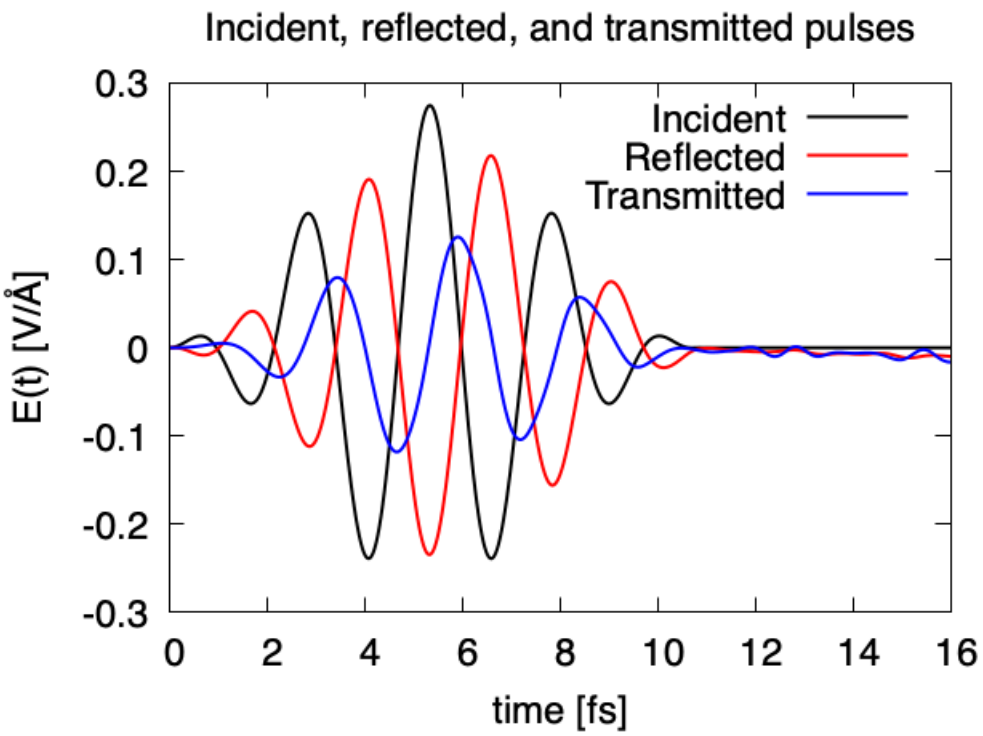
Explanations of other output files are given below:

Si_wave.data

Waveforms of incident, reflected, and transmitted waves.

```
# 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[fs] 2:E_inc_x[V/Angstrom] 3:E_inc_y[V/Angstrom] 4:E_inc_z[V/Angstrom]
# 5:E_ref_x[V/Angstrom] 6:E_ref_y[V/Angstrom] 7:E_ref_z[V/Angstrom] 8:E_tra_x[V/
↪Angstrom]
# 9:E_tra_y[V/Angstrom] 10:E_tra_z[V/Angstrom]
```

The figure below shows the incident, reflected, and transmitted electric fields that are drawn using the first column (time in femtosecond) and the 4th column (incident), 7th column (reflected), and 10th column (transmitted).



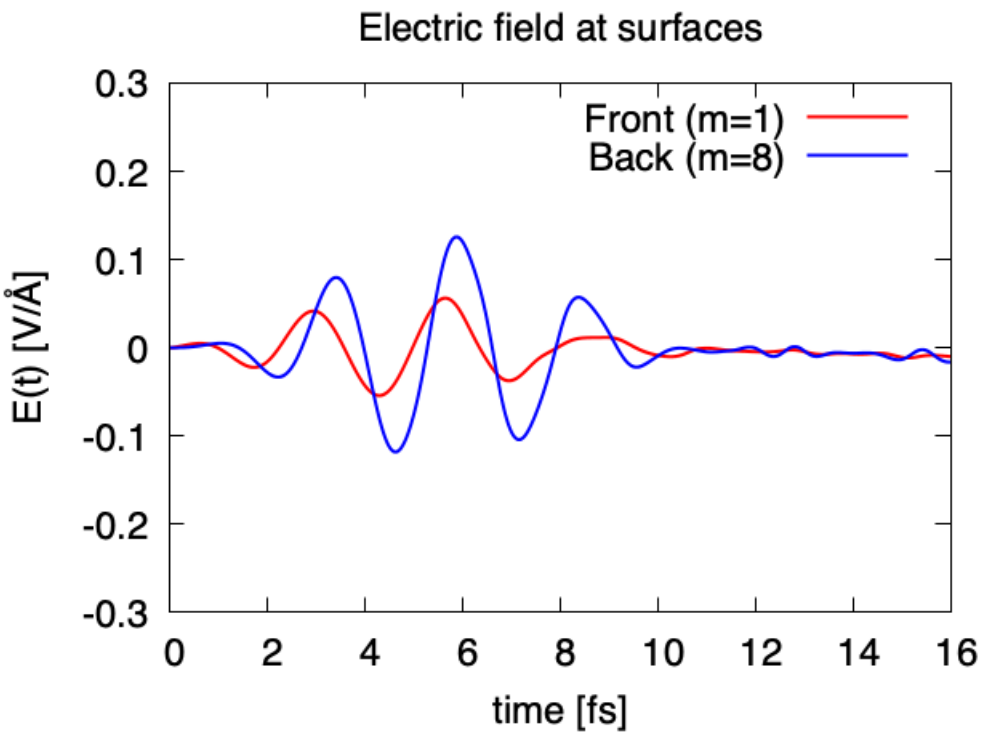
We find that the amplitude of the reflected pulse is comparable to the amplitude of the incudent pulse, while the phase is different by π. The amplitude of the transmitted pulse is smaller than the incident pulse.

Si_m/mxxxxxx/Si_rt.data

The number *xxxxxx* in the directory name *mxxxxxx* specifies the position of macroscopic grid point. Vector potential, electric field, and matter current density as functions of time are included in the file.

```
# 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[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:Jm_x[1/fs*Angstrom^2]
# 15:Jm_y[1/fs*Angstrom^2] 16:Jm_z[1/fs*Angstrom^2]
```

The figure below shows the electric field at front and back surfaces. Using 1st column (time in femtosecond) and 13th column (total electric field in z direction), electric field at a macroscopic poisition inside the thin film can be plotted. Using the file `/m000001/Si_rt.data`, electric field at the front surface is drawn by red curve. Using the file `/m000008/Si_rt.data`, electric field at the back surface is drawn by blue curve.



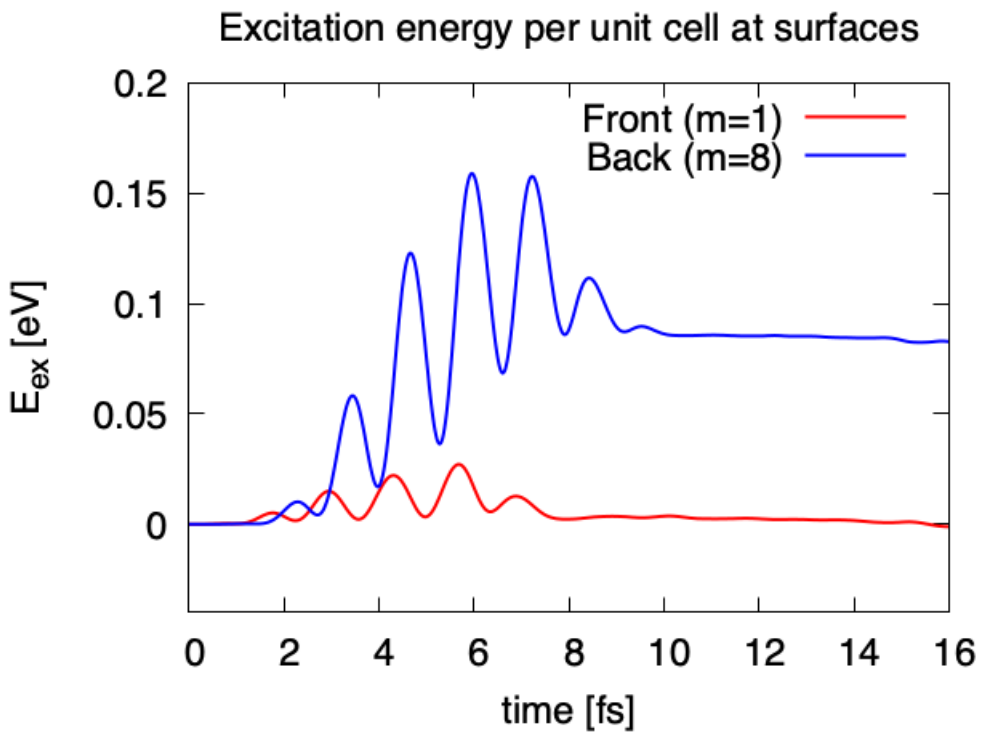
We find that the amplitude of the electric field at the front surface is small. It is consistent with the previous figure that showed incident and reflected pulses with a similar amplitude and opposite phase.

Si_m/mxxxxxx/Si_rt_energy.data

The number *xxxxxx* in the directory name *mxxxxxx* specifies the position of macroscopic grid point. *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]
```

The figure below shows the electronic excitation energy per unit cell volume at front and back surfaces using 1st columnn (time in femtosecond) and 3rd column (*Eall-Eall0*). Using the file `/m000001/Si_rt_energy.data`, the excitation energy at the front surface is drawn by red curve. Using the file `/m000008/Si_rt_energy.data`, the excitation energy at the back surface is drawn by blue curve.



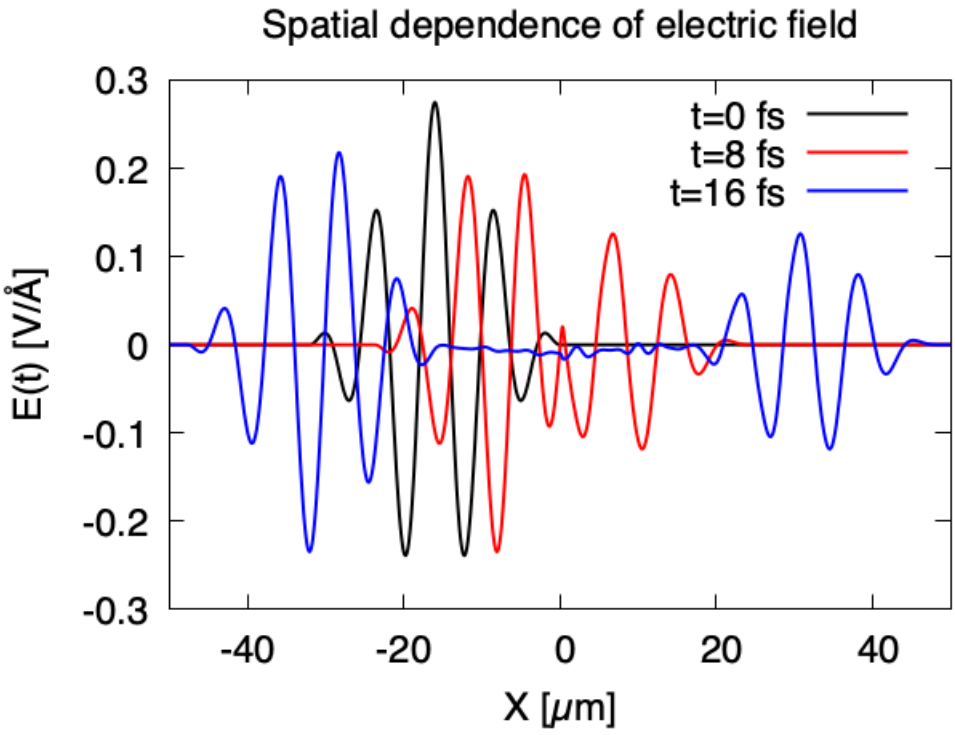
The excitation energy is much larger at the back surface compared with the energy at the front surface. This is because the amplitude of the electric field at the back surface is larger than that of the front surface, as seen in the previous figure, and the excitation is a nonlinear process.

Si_RT_Ac/Si_Ac_yyyyyy.data

The number *yyyyyy* in the file name `Si_Ac_yyyyyy.data` specifies the time step. Various quantities at the time step are included in the file as functions of macroscopic position index.

```
# 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[fs*V/Angstrom] 5:Ac_y[fs*V/Angstrom]
# 6:Ac_z[fs*V/Angstrom] 7:E_x[V/Angstrom] 8:E_y[V/Angstrom] 9:E_z[V/Angstrom] 10:B_
↪x[a.u.]
# 11:B_y[a.u.] 12:B_z[a.u.] 13:Jem_x[1/fs*Angstrom^2] 14:Jem_y[1/fs*Angstrom^2]
# 15:Jem_z[1/fs*Angstrom^2] 16:E_em[eV/vol] 17:E_abs[eV/vol]
```

The figure below shows spatial dependence of the electric field at three times, $t = 0$ fs (initial), $t = 8$ fs (pulse goes through the film), and $t = 16$ fs (final). It is drawn using the first column multiplied by the step size of X and 9th column (electric field).



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

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 input keywords*.

```
!#############################################################################
↪###!
! Excercise 08: Geometry optimization of C2H2 molecule                       ␣
↪    !
```

(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 input keywords' in the manual)                       ␣
↪   !
!-------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                          ␣
↪   !
!   Length: 1 [a.u.] = 0.52917721067    [Angstrom]                              ␣
↪   !
!   Energy: 1 [a.u.] = 27.21138505      [eV]                                    ␣
↪   !
!   Time  : 1 [a.u.] = 0.02418884326505 [fs]                                    ␣
↪   !
!###############################################################################
↪###!

&calculation
  !type of theory
  theory = 'dft'

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

theory specifies which theoretical method is used in the calculation.
yn_opt is a switch to carry out the structure optimization.

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

sysname is a prefix for filenames of output files.

```
&units
  !units used in input and output files
```

(continues on next page)

(continued from previous page)

```
  unit_system = 'A_eV_fs'
/
```

unit_system specifies which unit system is used in the input and output files.

```
&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 specifies whether or not periodic boundary condition is applied.
al(i) specifies the spatial box size of the cubiod cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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. !
  !------------------------------------------------!
/
```

file_pseudo(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

dl(i) specifies the spatial grid spacing in i-th direction.

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

nscf specifies the maximum number of SCF iterations.
threshold specifies the threshold to judge the convergence.

```
&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.    !
  !-------------------------------------------------------------------!
/
```

&atomic_coor specifies spatial coordinates of atoms.

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_1/exercise_zip_files/08_C2H2_opt.zip

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 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                                                  ␣
↪    !
!      /                                                                       ␣
↪    !
!    (see chapter: 'List of input keywords' in the manual)                     ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                         ␣
↪    !
!    Length: 1 [a.u.] = 0.52917721067    [Angstrom]                            ␣
↪    !
!    Energy: 1 [a.u.] = 27.21138505      [eV]                                  ␣
↪    !
!    Time  : 1 [a.u.] = 0.02418884326505 [fs]                                  ␣
↪    !
```

(continues on next page)

(continued from previous page)

```
!-------------------------------------------------------------------------------
↪---!
! * 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'
/
```

theory specifies which theoretical method is used in the calculation.
yn_md is a switch for Ehrenfest molecular dynamics.

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

sysname is a prefix for filenames of output files.

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

unit_system specifies which unit system is used in the input and output files.

```
&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)
```

(continues on next page)

(continued from previous page)

```
  nelem  = 2
  natom  = 4
  nelec  = 10
  nstate = 6
/
```

yn_periodic specifies whether or not periodic boundary condition is applied.
al(i) specifies the spatial box size of the cubiod cell.
nelem is the number of elements in the system.
natom is the number of atoms in the system.
nelec is the number of electrons in the system.
nstate is the number of orbitals that are used in the calculation.

```
&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. !
  !------------------------------------------------!
/
```

file_pseudo(n) specifies the filename of the pseudopotential file of the n-th element.
izatom(n) is the atomic number of the n-th element.
lloc_ps(n) specifies which angular momentum component is chosen as the local potential for the n-th element.

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

xc specifies the exchange-correlation potential to be used in the calculation.

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

dl(i) specifies the spatial grid spacing in i-th direction.

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

dt specifies the time step.
nt is the number of time steps for the time propagation.

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

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.
phi_cep1 specifies the carrier-envelope phase of the pulse.

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

(continues on next page)

(continued from previous page)

```
  !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
/
```

ensemble specifies the choice of the ensemble.
yn_set_ini_velocity is a switch to prepare initial velocity for atoms.
temperature0_ion_k specifies the temperature that is used to generate initial velocity of ions.
step_update_ps specifies the time step interval to update projector for the nonlocal 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_1/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
```

(continues on next page)

(continued from previous page)

```
# 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: Absorption-, Scattering-, and Extinction-cross-sections of an Au nanoparticle in FDTD simulation

In this exercise, we learn the calculation of the absorption-, scattering-, and extinction-cross-sections of an Au nanoparticle, by applying the incident pulse in the time-dependent Maxwell equations. As outputs of the calculation, those cross-sections and the time response of the electromagnetic field are 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 `AuNP_fdtd.inp` is used:

file name	description
AuNP_fdtd.inp	input file that contains input keywords and their values.

In the input file `AuNP_fdtd.inp`, input keywords are specified. Most of them are mandatory to execute this exercise. 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 input keywords*.

```
!################################################################################
↪###!
! Excercise 10: Absorption-, Scattering-, and Extinction-cross-sections       ␣
↪    !
!               of an Au nanoparticle in FDTD simulation                       ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                      ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':                         ␣
↪    !
!   Length: 1 [a.u.] = 0.52917721067    [Angstrom]                             ␣
↪    !
!   Energy: 1 [a.u.] = 27.21138505      [eV]                                   ␣
↪    !
!   Time  : 1 [a.u.] = 0.02418884326505 [fs]                                   ␣
↪    !
!################################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

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

sysname is a prefix for filenames of output files.
base_directory specifies the directory name where output files are generated.

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

unit_system specifies which unit system is used in the input and output files.

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

yn_periodic specifies whether or not periodic boundary condition is applied.

```
&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 = 1.00d6

  !duration of the incident pulse
  tw1 = 7.50d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 2.30d0

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

  !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 ------------------------------------------------------!
```

(continues on next page)

(continued from previous page)

```
  ! Definition of the incident pulse is written in:                    !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !---------------------------------------------------------------------!
/
```

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.
phi_cep1 specifies the carrier-envelope phase of the pulse.

```
&maxwell
  !grid box size(x,y,z) and number of spatial grids(x,y,z)
  al_em(1:3)        = 600.0d1, 600.0d1, 600.0d1
  num_rgrid_em(1:3) = 100, 100, 100
  !--- Caution -------------------------------------------------------!
  ! Copumutational domain is set as:                                  !
  !         -al_em/2 ~ al_em/2 for yn_periodic='n'                    !
  ! whereas        0 ~ al_em   for yn_periodic='y'.                   !
  !---------------------------------------------------------------------!

  !total time
  at_em = 50.0d0
  !--- TIPS ----------------------------------------------------------!
  ! Two of at_em, dt_em, and nt_em must be set.                       !
  ! Otherwise, both at_em and nt_em or either of those must be set.   !
  ! The latter automatically determines dt_em from CFL condition.     !
  !---------------------------------------------------------------------!

  !*** SHAPE INFORMATION(START) ***************************************!
  !make and output shape file
  yn_make_shape   = 'y'
  yn_output_shape = 'y'

  !number of shape-template
  n_s = 1

  !media ID and type of shape-template(shape ID)
  id_s(1)  = 1
  typ_s(1) = 'ellipsoid'

  !information and origin of shape-template:
  !inf_s(shape ID,x-diameter,y-diameter,z-diameter)
  !ori_s(shape ID,x,y,z)
  inf_s(1,1:3) = 200.0d1, 200.0d1, 200.0d1
  ori_s(1,1:3) = 0.000d1, 0.000d1, 0.000d1
  !--- TIPS ----------------------------------------------------------!
  ! * shape file can be generated by also an external program         !
  !   'FDTD_make_shape' in SALMON utilities                           !
  !   (https://salmon-tddft.jp/utilities.html).                       !
  !   The generated shape file can be read by an input keyword        !
```

(continues on next page)

(continued from previous page)

```
!   'shape_file' in &maxwell.                                              !
! * More complex shapes can be generated by other input keywords          !
!   which are in common with those used in 'FDTD_make_shape'.              !
!-------------------------------------------------------------------------!
!*** SHAPE INFORMATION(END) ***************************************!

!*** MEDIA INFORMATION(START) **************************************!
!number and type of media(media ID)
media_num     = 1
media_type(1) = 'lorentz-drude'
!--- Au described by Lorentz-Drude model -----------------------------!
! The parameters are determined from:                                   !
! (https://doi.org/10.1364/AO.37.005271)                                !
!-------------------------------------------------------------------------!

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

!oscillator strength, collision frequency,
!and oscillator frequency of LD 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
!--- TIPS ------------------------------------------------------------!
! If you calclate a metallic nanoparticle surrounded by something      !
! other than vacuum/air, such as Au nanoparticle in water solution,    !
! you should change 'epsilon_em(0)' in &maxwell,                       !
! which specifies the permittivity of surrounding medium.              !
!-------------------------------------------------------------------------!
!*** 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) = 0.000d1, 0.000d1, -200.0d1

!propagation direction of the incident pulse(x,y,z)
ek_dir1(1:3) = 0.000d0, 0.000d0, 1.000d0
!*** SOURCE INFORMATION(END) ***************************************!

!*** ASE INFORMATION(START) ****************************************!
!number of wavelength grid points
!for Absorption-, Scattering-, and Extinction-cross-sections(ASE)
!normalized by the spectral distribution of the incident pulse
ase_num_em = 100

!minimum and maximum values of wavelength for ASE
ase_wav_min_em = 400.0d1
ase_wav_max_em = 800.0d1
!--- TIPS ------------------------------------------------------------!
! ase_ene_min_em and ase_wav_max_em can also be available.             !
! If those are used, ASE are outputed for energy axis.                 !
!-------------------------------------------------------------------------!
```

(continues on next page)

(continued from previous page)

```
  !size of a closed surface (box shape) to calculate ASE(x-size,y-size,z-size)
  ase_box_size_em(1:3) = 300.0d1, 300.0d1, 300.0d1
  !*** ASE INFORMATION(END) ***************************************!

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

  !location of observation point(observation ID,x,y,z)
  obs_loc_em(1,1:3) = 0.0d0, 0.0d0, 0.0d0
  !--- TIPS ----------------------------------------------------------!
  ! * If you specify yn_obs_plane_em(1) = 'y',                       !
  !   animation files can be made by an external program             !
  !   'FDTD_make_figani' in SALMON utilities.                        !
  !   (https://salmon-tddft.jp/utilities.html).                      !
  !   The animation file visualizes electromagnetic field distributions !
  !   on the cross-section including the observation point           !
  !   whose location is determined by obs_loc_em.                    !
  ! * If you specify obs_plane_ene_em(1,1:n) by certain values       !
  !   space-energy distribution of electromagnetic field is outputed !
  !-------------------------------------------------------------------!
  !*** OBSERVATION INFORMATION(END) **********************************!
/
```

al_em(i) specifies the lengths of three sides of the cuboid where the grid points are prepared.
num_rgrid_em(i) specifies the number of grid points in i-th direction.
at_em specifies total time for electromagnetic analysis.
yn_make_shape is a switch for making shape. This is same functionality for `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
yn_output_shape is a switch for outputing shape in cube file format.
n_s specifies number of shape-template.
id_s(n) specifies media ID for n-th shape-template.
typ_s(n) specifies type for n-th shape-template.
inf_s(n,i) specifies i-th information for n-th shape-template.
ori_s(n,i) specifies origin for n-th shape-template.
media_num specifies the number of the types of media that is provided in the shape file.
media_type(n) specifies the type of the n-th media.
pole_num_ld(n) and *omega_p_ld(n)* specify the number of poles and the plasmal frequency of the n-th media, respectively.
f_ld(n,m), *omega_ld(n,m)*, *gamma_ld(n,m)* specify the oscillator strength, oscillator frequency, and collision frequency of the m-th pole of the n-th media, respectively.
wave_input specifies an electric current source that is used for the generation of the pulse.
source_loc1(i) specifies the coordinate of the current source.
ek_dir1(i) specifies the propagation direction of the pulse.
ase_num_em specifies number of wavelength grid points for Absorption-, Scattering-, and Extinction-cross-sections(ASE).
ase_wav_min_em specifies minimum value of wavelength for ASE.
ase_wav_max_em specifies maximum value of wavelength for ASE.
ase_box_size_em specifies size of a closed surface (box shape) to calculate ASE.

obs_num_em specifies the number of the observing point.
obs_loc_em(n,i) specifies the coordinate of n-th observing point.

Output files

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

file name	description
AuNP_ase_with_wf.data	A-, S-, and E-corss-sections as functions of wavelength, where window function is applied in Fourier transformation
AuNP_ase_without_wf.data	A-, S-, and E-corss-sections as functions of wavelength, where window function is not applied in Fourier transformation
obs1_at_point_rt.data	components of electric and magnetic fields as functions of time
shape.cube	shape file for fdtd

Explanations of the files are described below:

AuNP_ase_with_wf.data

Results of A-, S-, and E-cross-sections normalized by the spectral distribution of the incident pulse. Also the spectral distribution for pointing vector of incident pulse is included. Window function is applied in Fourier transformation.

```
# Absorption-, Scattering-, and Extinction-cross-sections normalized by the spectral␣
↪distribution of the incident pulse (with window function):
# sigma_a: Absorption cross-section
# sigma_s: Scattering cross-section
# sigma_e: Extinction cross-section
# S: Pointing vector of incident pulse along propagation direction
# 1:Wavelength[Angstrom] 2:sigma_a[Angstrom^2] 3:sigma_s[Angstrom^2] 4:sigma_
↪e[Angstrom^2] 5:S[VA/Angstrom^2*fs^2]
```

AuNP_ase_without_wf.data

Results of A-, S-, and E-cross-sections normalized by the spectral distribution of the incident pulse. Also the spectral distribution for pointing vector of incident pulse is included. Window function is not applied in Fourier transformation.

```
# Absorption-, Scattering-, and Extinction-cross-sections normalized by the spectral␣
↪distribution of the incident pulse (without window function):
# sigma_a: Absorption cross-section
# sigma_s: Scattering cross-section
# sigma_e: Extinction cross-section
# S: Pointing vector of incident pulse along propagation direction
# 1:Wavelength[Angstrom] 2:sigma_a[Angstrom^2] 3:sigma_s[Angstrom^2] 4:sigma_
↪e[Angstrom^2] 5:S[VA/Angstrom^2*fs^2]
```

obs0_info.data

This file is used to generate animation files by using SALMON utilities with *yn_obs_plane_em*: 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
```

(continues on next page)

(continued from previous page)

```
# 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]
```

shape.cube

Shape file generated by *yn_make_shape*.

3.6.2 Exercise-11: Absorption-, Reflection-, and Transmission-rates of an Au nanoparticles metasurface in FDTD simulation

In this exercise, we learn the calculation ofthe absorption-, reflection-, and transmission-rates of a metasurface, in which Au nanoparticles are periodically arrayed in two-dimension, by applying the incident pulse in the time-dependent Maxwell equations. As outputs of the calculation, those rates and the time response of the electromagnetic field are 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 `AuNP_fdtd.inp` is used:

file name	description
AuNPs_fdtd.inp	input file that contains input keywords and their values.

In the input file `AuNPs_fdtd.inp`, input keywords are specified. Most of them are mandatory to execute this exercise. 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 input keywords*.

```
!################################################################################
↪###!
! Excercise 11: Absorption-, Reflection-, and Transmission-rates                 ␣
↪    !
!               of an Au nanoparticles metasurface in FDTD simulation           ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
! * The detail of this excercise is explained 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 input keywords' in the manual)                       ␣
↪    !
!-------------------------------------------------------------------------------
↪---!
```

(continues on next page)

(continued from previous page)

```
! * Conversion from unit_system = 'a.u.' to 'A_eV_fs':
↪   !
!   Length: 1 [a.u.] = 0.52917721067    [Angstrom]
↪   !
!   Energy: 1 [a.u.] = 27.21138505      [eV]
↪   !
!   Time  : 1 [a.u.] = 0.02418884326505 [fs]
↪   !
!#############################################################################
↪###!

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

theory specifies which theoretical method is used in the calculation.

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

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

sysname is a prefix for filenames of output files.
base_directory specifies the directory name where output files are generated.

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

unit_system specifies which unit system is used in the input and output files.

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

yn_periodic specifies whether or not periodic boundary condition is applied.

```
&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 = 1.00d6

  !duration of the incident pulse
  tw1 = 7.50d0

  !mean photon energy(average frequency multiplied by the Planck constant) of the␣
↪incident pulse
  omega1 = 2.30d0

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

  !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 ------------------------------------------------------!
  ! Definition of the incident pulse is written in:                   !
  ! https://www.sciencedirect.com/science/article/pii/S0010465518303412 !
  !--------------------------------------------------------------------!
/
```

ae_shape1 specifies the envelope of the field.
I_wcm2_1 specify the intensity of the pulse in unit of W/cm^2.
tw1 specifies the duration of the pulse.
omega1 specifies the mean photon energy of the pulse.
epdir_re1(i) specifies the i-th component of the real part of the polarization unit vector.
phi_cep1 specifies the carrier-envelope phase of the pulse.

```
&maxwell
  !grid box size(x,y,z) and number of spatial grids(x,y,z)
  al_em(1:3)        = 300.0d1, 300.0d1, 600.0d1
  num_rgrid_em(1:3) = 50, 50, 100
  !--- Caution ------------------------------------------------------!
  ! Copumutational domain is set as:                                  !
  !         -al_em/2 ~ al_em/2 for yn_periodic='n'                    !
  ! whereas        0 ~ al_em   for yn_periodic='y'.                   !
  !--------------------------------------------------------------------!

  !total time
  at_em = 50.0d0
  !--- TIPS ---------------------------------------------------------!
  ! Two of at_em, dt_em, and nt_em must be set.                       !
  ! Otherwise, both at_em and nt_em or either of those must be set.   !
  ! The latter automatically determines dt_em from CFL condition.     !
  !--------------------------------------------------------------------!

  !absorbing boudnary conditions at bottom and top on z axis
```

(continues on next page)

(continued from previous page)

```
boundary_em(3,1) = 'abc'
boundary_em(3,2) = 'abc'

!*** SHAPE INFORMATION(START) ***************************************!
!make and output shape file
yn_make_shape   = 'y'
yn_output_shape = 'y'

!number of shape-template
n_s = 1

!media ID and type of shape-template(shape ID)
id_s(1)  = 1
typ_s(1) = 'ellipsoid'

!information and origin of shape-template:
!inf_s(shape ID,x-diameter,y-diameter,z-diameter)
!ori_s(shape ID,x,y,z)
inf_s(1,1:3) = 200.0d1, 200.0d1, 200.0d1
ori_s(1,1:3) = 150.0d1, 150.0d1, 300.0d1
!--- TIPS ----------------------------------------------------------!
! * shape file can be generated by also an external program          !
!   'FDTD_make_shape' in SALMON utilities                            !
!   (https://salmon-tddft.jp/utilities.html).                        !
!   The generated shape file can be read by an input keyword         !
!   'shape_file' in &maxwell.                                        !
! * More complex shapes can be generated by other input keywords     !
!   which are in common with those used in 'FDTD_make_shape'.        !
!-------------------------------------------------------------------!
!*** SHAPE INFORMATION(END) *****************************************!

!*** MEDIA INFORMATION(START) ***************************************!
!number and type of media(media ID)
media_num     = 1
media_type(1) = 'lorentz-drude'
!--- Au described by Lorentz-Drude model ---------------------------!
! The parameters are determined from:                               !
! (https://doi.org/10.1364/AO.37.005271)                           !
!-------------------------------------------------------------------!

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

!oscillator strength, collision frequency,
!and oscillator frequency of LD 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
!--- TIPS ----------------------------------------------------------!
! If you calclate a metallic nanoparticles surrounded by something  !
! other than vacuum/air, such as Au nanoparticles in water solution,!
! you should change 'epsilon_em(0)' in &maxwell,                    !
! which specifies the permittivity of surrounding medium.           !
!-------------------------------------------------------------------!
!*** MEDIA INFORMATION(END) *****************************************!
```

(continues on next page)

(continued from previous page)

```
  !*** 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) = 0.000d1, 0.000d1, 100.0d1

  !propagation direction of the incident pulse(x,y,z)
  ek_dir1(1:3) = 0.000d0, 0.000d0, 1.000d0
  !*** SOURCE INFORMATION(END) **************************************!

  !*** ART INFORMATION(START) ***************************************!
  !number of wavelength grid points
  !for Absorption-, Reflection-, and Transmission-rates(ART)
  !normalized by the spectral distribution of the incident pulse
  art_num_em = 100

  !minimum and maximum values of wavelength for ART
  art_wav_min_em = 400.0d1
  art_wav_max_em = 800.0d1
  !--- TIPS ---------------------------------------------------------!
  ! art_ene_min_em and art_wav_max_em also can be available.          !
  ! If those are used, ART are outputed for energy axis.              !
  !--------------------------------------------------------------------!

  !location of bottom and top planes on the propagation axis to calculate ART(x,y,z)
  art_plane_bot_em(1:3) = 150.0d1, 150.0d1, 150.0d1
  art_plane_top_em(1:3) = 150.0d1, 150.0d1, 450.0d1
  !*** ART INFORMATION(END) *****************************************!

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

  !location of observation point(observation ID,x,y,z)
  obs_loc_em(1,1:3) = 150.0d1, 150.0d1, 300.0d1
  !--- TIPS ---------------------------------------------------------!
  ! * If you specify yn_obs_plane_em(1) = 'y',                       !
  !   animation files can be made by an external program             !
  !   'FDTD_make_figani' in SALMON utilities.                        !
  !   (https://salmon-tddft.jp/utilities.html).                      !
  !   The animation file visualizes electromagnetic field distributions !
  !   on the cross-section including the observation point           !
  !   whose location is determined by obs_loc_em.                    !
  ! * If you specify obs_plane_ene_em(1,1:n) by certain values       !
  !   space-energy distribution of electromagnetic field is outputed !
  !--------------------------------------------------------------------!
  !*** OBSERVATION INFORMATION(END) **********************************!
/
```

al_em(i) specifies the lengths of three sides of the cuboid where the grid points are prepared.
num_rgrid_em(i) specifies the number of grid points in i-th direction.
at_em specifies total time for electromagnetic analysis.
boundary_em(i,n) specifies the boundary condition for the electromagnetic analysis. The first index i corresponds to

the x,y, and z direction. The second index n specifies bottom or top of the material.
yn_make_shape is a switch for making shape. This is same functionality for `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
yn_output_shape is a switch for outputing shape in cube file format.
n_s specifies number of shape-template.
id_s(n) specifies media ID for n-th shape-template.
typ_s(n) specifies type for n-th shape-template.
inf_s(n,i) specifies i-th information for n-th shape-template.
ori_s(n,i) specifies origin for n-th shape-template.
media_num specifies the number of the types of media that is provided in the shape file.
media_type(n) specifies the type of the n-th media.
pole_num_ld(n) and *omega_p_ld(n)* specify the number of poles and the plasmal frequency of the n-th media, respectively.
f_ld(n,m), *omega_ld(n,m)*, *gamma_ld(n,m)* specify the oscillator strength, oscillator frequency, and collision frequency of the m-th pole of the n-th media, respectively.
wave_input specifies an electric current source that is used for the generation of the pulse.
source_loc1(i) specifies the coordinate of the current source.
ek_dir1(i) specifies the propagation direction of the pulse.

art_num_em specifies number of wavelength grid points for Absorption-, Reflection-, and Transmission-rates(ART).
art_wav_min_em specifies minimum value of wavelength for ART.
art_wav_max_em specifies maximum value of wavelength for ART.
art_plane_bot_em specifies location of bottom plane on the propagation axis to calculate ART
art_plane_top_em specifies location of top plane on the propagation axis to calculate ART
obs_num_em specifies the number of the observing point.
obs_loc_em(n,i) specifies the coordinate of n-th observing point.

Output files

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

file name	description
AuNPs_art_with_wf.data	A-, R-, and T-rates as functions of wavelength, where window function is applied in Fourier transformation
AuNPs_art_without_wf.data	A-, R-, and T-rates as functions of wavelength, where window function is not applied in Fourier transformation
obs1_at_point_rt.data	components of electric and magnetic fields as functions of time
shape.cube	shape file for fdtd

Explanations of the files are described below:

AuNPs_art_with_wf.data

Results of A-, R-, and T-rates normalized by the spectral distribution of the incident pulse. Also the spectral distribution for pointing vector of incident pulse is included. Window function is applied in Fourier transformation.

```
# Absorption-, Reflection-, and Transmission-rates normalized by the spectral␣
↪distribution of the incident pulse (with window function):
# A: Absorption rate
```

(continues on next page)

(continued from previous page)

```
# R: Reflection rate
# T: Transmission rate
# S: Pointing vector of incident pulse along propagation direction
# 1:Wavelength[Angstrom] 2:A % 3:R % 4:T % 5:S[VA/Angstrom^2*fs^2]
```

AuNPs_art_without_wf.data

Results of A-, R-, and T-rates normalized by the spectral distribution of the incident pulse. Also the spectral distribution for pointing vector of incident pulse is included. Window function is not applied in Fourier transformation.

```
# Absorption-, Reflection-, and Transmission-rates normalized by the spectral␣
↪distribution of the incident pulse (without window function):
# A: Absorption rate
# R: Reflection rate
# T: Transmission rate
# S: Pointing vector of incident pulse along propagation direction
# 1:Wavelength[Angstrom] 2:A % 3:R % 4:T % 5:S[VA/Angstrom^2*fs^2]
```

obs0_info.data

This file is used to generate animation files by using SALMON utilities with *yn_obs_plane_em*: 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]
```

shape.cube

Shape file generated by *yn_make_shape*.

3.7 [Trial] Semiconductor Bloch equation

The SALMON program includes a time evolution calculation feature based on the Semiconductor Bloch Equation (SBE), which was added starting from version 2.2.0. The SBE calculation feature is currently an experimental implementation, and the developers cannot guarantee its behavior. To activate the SBE calculation feature, specify the value `sbe` or `sbe_maxwell` for the *theory* option in the input file; these feature includes both a real-time calculation feature equivalent to "tddft_pulse" and a multiscale calculation feature equivalent to "multi_scale_maxwell_tddft" that are already available in the program.

3.7.1 [Trial] Exercise-x1: Semiconductor Bloch equation (SBE) calculation

To run the code, following files in the directory `SALMON/samples/exercise_x1_bulkSi_sbe_gs_rt/` are used:

file name	description
Si_gs.inp	input file for ground state calculation keywords and their values
Si_rps.dat	pseodupotential file for silicon atom
Si_sbe_rt.inp	input file for real time calculation keywords and their values

Ground state calculation

Before performing SBE calculations, various information of ground state such as the eigenenergy and transition dipole moment imust be prepared. The conditions for the ground state calculation are specified in `Si_gs.inp`; please refer to *Exercise-4: Ground state of crystalline silicon* for more details. Note that it is necessary to set `yn_out_tm` parameter to output the transition dipole moment.

```
&analysis
   yn_out_tm = "y"
/
```

Output files of ground state calculation

After the calculation, a few output files are created; on particular, the following three files are important for SBE calculations.

file name	description
Si_k.data	sampled k-point coordinates in BZ
Si_eigen.data	Eigenenergies
Si_tm.data	Transition dipole moment matrix

Real-time SBE calculation

To perform real-time calculations, it is necessary to place the three data files from the ground-state calculation, `SYSNAME_k.data`, `SYSNAME_eigen.data`, and `SYSNAME_tm.data`, in the same directory. If the real-time calculation is performed in a different directory from the ground-state calculation, it is necessary to manually copy (or link) the above files.

```
&calculation
   theory = 'sbe'
/
```

The parameter `theory='sbe'` must be specified.

```
&control
   sysname = 'Si'
/

&units
   unit_system = 'au'
/

&system
   yn_periodic = 'y'
   al(1:3) = 10.26d0, 10.26d0, 10.26d0
   nelem = 1
```

(continues on next page)

(continued from previous page)

```
   natom = 8
   nelec = 32
   nstate = 32
/

&kgrid
   num_kgrid(1:3) = 8, 8, 8
/

&tgrid
   dt = 0.05d0
   nt = 20000
/

&emfield
   ae_shape1 = "Acos2"
   epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
   I_wcm2_1 = 1.0d+12
   tw1 = 1000.0d0
   omega1 = 0.056d0
/
```

The above section shares the same parameters as the time-evolution calculations for bulk crystals. please refer to *Exercise-6: Electron dynamics in crystalline silicon under a pulsed electric field* for more details.

3.7.2 [Trial] Exercise-x2: Multiscale Maxwell semiconductor Bloch equation (Maxwell+SBE) calculation

To run the code, following files in the directory SALMON/samples/exercise_x2_bulkSi_bloch_gs_ms/ are used:

file name	description
Si_gs.inp	Inputfile for ground state calculation
Si_sbe_ms_1d_film.inp	Inputfile for 1D multiscale calculation
Si_sbe_ms_2d_cylinder.inp	Inputfile for 2D multiscale calculation

To perform Maxwell+SBE multiscale calculations, various data of the ground state, such as eigenenergy and transition dipole moment, are required. It is necessary to perform the ground-state calculation: `Si_gs.inp` before executing multiscale calculations. See *Exercise-x1* .

Multiscale calculation for laser pulse propagation in silicon nano-film (1D calculation)

***Si_sbe_ms_1d_film.inp**

```
&calculation
   theory = 'maxwell_sbe'
/
```

The parameter `theory='maxwell_sbe'` must be specified.

```
&system
   yn_periodic = 'y'
   al(1:3) = 10.26d0, 10.26d0, 10.26d0
```

(continues on next page)

(continued from previous page)

```
   nelem = 1
   natom = 8
   nelec = 32
   nstate = 32
/

&emfield
   ae_shape1 = "Acos2"
   epdir_re1(1:3) = 0.0d0, 0.0d0, 1.0d0
   I_wcm2_1 = 1.0d+12
   tw1 = 1000.0d0
   omega1 = 0.056d0
/

&multiscale
   nx_m = 20
   ny_m = 1
   nz_m = 1
   hx_m = 94.52 ! 5nm
   hy_m = 94.52 ! 5nm
   hz_m = 94.52 ! 5nm
   nxvac_m(1) = 2000
   nxvac_m(2) = 2000
/
```

The above section shares the same parameters as the time-evolution calculations for bulk crystals. Please refer to *Exercise-7: Pulsed-light propagation through a silicon thin film* for more details.

Multiscale calculation for laser pulse incident on arbitrary-shaped nanostructures (2D calculation)

The multidimensional multiscale method (2D or 3D) can handle the light-matter interaction with arbitrarily shaped nanostructures. As a example, the periodically arranged silicon nanocylinder array is considered

***Si_sbe_ms_2d_cylinder.inp**

Most parts of the input file are common with the previous 1D calculatio.

```
&multiscale
   fdtddim = "3d"
   hx_m = 189.0 ! 5nm
   hy_m = 189.0 ! 5nm
   hz_m = 189.0 ! 5nm
   nx_m = 80
   ny_m = 80
   nz_m = 1
   nxvac_m(1) = 2000
   nxvac_m(2) = 2000
/
```

This section defines a 2D computational domain of 80 cells x 80 cells (400 nm x 400 nm). Furthermore, a vacuum region of 2000 cells (10 um) is added along the x-axis to surround the computational domain.

```
&maxwell
   ! Media 1
   media_type(1) = "multiscale"
   ! Shaper
```

(continues on next page)

(continued from previous page)

```
   n_s = 1
   ! Object 1
   id_s(1) = 1
   typ_s(1) = "ellipsoid"
   ori_s(1,1:3) = 7561.43, 7561.43, 94.52
   inf_s(1,1:3) =  7561.43,  7561.43, 10000.0
   ! Detectors
   obs_num_em=3
   obs_loc_em(1, 1:3) = 7561.43, 7561.43, 94.52
   obs_loc_em(2, 1:3) = 11342.145, 7561.43, 94.52
   obs_loc_em(3, 1:3) = 15122.86, 7561.43, 94.52
/
```

This section defines the shape, coordinates and arrangement of the macroscopic objects. The most of parameters are common with `theory='maxwell'`. Please refer to *Exercise-10* for more details.

Note that if you want to treat the medium with electron dynamics calculations (TDDFT or SBE), specify‘‘media_type‘‘ as `'multiscale'`.

CHAPTER 4

List of input keywords

'[Trial]' : These options are not tested well

4.1 &calculation

4.1.1 theory

character, default=''

Choice of a theory to be used in the calculation.
Options:

`dft` / ground state calculation based on DFT
`dft_md` / ab initio MD simulations based on DFT (electronic ground state)
`tddft_response` / linear response TDDFT calculation in real time
`tddft_pulse` / simulations under pulsed electric field based on TDDFT
`single_scale_maxwell_tddft` / single-scale simulation coupling Maxwell and TDDFT
`multi_scale_maxwell_tddft` / multiscale simulation coupling Maxwell and TDDFT
`maxwell` / electromagnetic analysis using finite difference time domain (FDTD) method
`dft_k_expand` / convert checkpoint data of dft with k-points calculation to that of larger supercell system with gamma-point
`sbe` / [Trial] simulations under pulsed electric field based in semiconductor Bloch equation
`maxwell_sbe` / [Trial] multiscale simulation coupling Maxwell and semiconductor Bloch equation

4.1.2 yn_md

[Trial] character, default='n'

Available for `theory='dft'` (ground-state MD) and `theory='tddft_pulse'` (Ehrenfest MD).
Switch for molecular dynamics calculation.

Options:

'y' / enable

'n' / disable

4.1.3 yn_opt

[Trial] character, default='n'

Available for theory='dft'.

Switch for geometry optimization.

Options:

'y' / enable

'n' / disable

4.2 &control

4.2.1 sysname

character, default='default'

Available for all options of theory.

A prefix of output files.

4.2.2 base_directory

character, default='./'

Available for all options of theory.

Name of a directory where major output files are stored.

4.2.3 yn_restart

character, default='n'

Available for the DFT/TDDFT based options of theory and theory='maxwell'.

Whether to continue previous calculation (restart) or start a new calculation.

Options:

'y' / enable (restart)

'n' / disable (new calculation)

4.2.4 directory_read_data

character, default='restart/'

Available for yn_restart='y'.

Directory name to read data that are required in the present calculation (restart) and were generated in previous calculations. For TDDFT based options, it specifies the name of the directory containing ground state results that were stored in 'data_for_restart'. When restarting from a checkpoint, it specifies the name of the directory that contains the checkpoint data.

4.2.5 yn_self_checkpoint

character, default='n'

Available for the DFT/TDDFT based options of `theory`.

With this option, each process writes/reads the restart (and checkpoint) data independently (self data format) so that the restart cost is reduced for large systems. Note that the number of processes and their assignments must be unchanged in restarting. The data is written out into 'checkpoint_gs_XXXXXX/' (DFT) or 'checkpoint_rt_XXXXXX/' (TDDFT).

Options:

`'y'` / enable

`'n'` / disable

4.2.6 checkpoint_interval

integer, default=-1

Available for the DFT/TDDFT based options of `theory` and `theory='maxwell'`.

Interval of time steps (iteration steps) to write down the checkpoint data during the time-propagation (SCF iteration). Checkpoint data will not be written if a negative value is set.

4.2.7 yn_reset_step_restart

character, default='n'

Available for `yn_restart='y'` in the DFT/TDDFT based options of `theory`.

With this option, the counter of the SCF iteration step (for DFT) or the counter of the time propagation step (for TDDFT) is reset to 0 at the restart. In the SCF iteration, the density data in the previous SCF iteration step are abondoned.

4.2.8 read_gs_restart_data

character, default='all'

Available for `yn_restart='y'` with `theory='dft'`.

Specify which data are read in the restart. Specified data that are generated in the previous calculation and are contained in the restart (or checkpoint) directory are used in restarting the SCF iteration of DFT. The default option `'all'` indicates the complete restart. In other options, a part of restart data are used (other data are prepared in the same way as in the initial SCF step).

Options:

`all` / all of restart data are read

`all:single` / same as `all` option but the data is read in the single file format even though the self data format is specified with `yn_self_checkpoint='y'` (i.e., the restart data is read in the single file format while written out in the self format)

`rho_inout` / only electron densities including those of previous iteration steps are read (from rho_inout.bin file)

`rho_inout:single` / same as `rho_inout` option but the data is read in the single file format even though the self data format is specified with `yn_self_checkpoint='y'`

`rho` / only the latest electron density is read (from user-made data)

`wfn` / only orbital wavefunctions are read

4.2.9 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 those of previous iteration steps are written out

`wfn` / only orbital wavefunctions are written out

`checkpoint_only` / the restart data are outputted only in the self data format (separated data for each process) at the last step into 'checkpoint_gs_XXXXXX/' directory (`yn_self_checkpoint='y'` is required) without generating the restart data into 'data_for_restart/' directory in the single file format.

Output data files are written out in the restart (or checkpoint) directory.

The default option `'all'` gives the complete set of restart data.

4.2.10 time_shutdown

[Trial] real(8), default=-1d0

Available for the DFT/TDDFT based options of `theory`.

Timer for automatic shutdown. The unit is second.

If a negative time is set, the automatic shutdown will not be performed.

4.2.11 method_wf_distributor

character, default='single'

Available for the DFT/TDDFT based options of `theory`.

A method to save/load orbital wavefunctions.

Options

`single`: all orbital wavefunctions are saved(loaded) to(from) a single file.

`slice` : each orbital wavefunction is saved(loaded) to(from) a file. This choice reduces the I/O costs, and increase the flexiblility in handling files for large systems.

4.2.12 nblock_wf_distribute

integer, default='16'

Available for `method_wf_distributor='slice'`.

In the 'slice' mode, `nblock_wf_distribute` files are saved in one directory.

4.3 &units

4.3.1 unit_system

character, default='au'

Unit system to be used in input variables and some of output files.

If `unit_system = 'A_eV_fs'` is chosen, Angstrom for length, eV for energy, and fs for time are adopted.

For isolated systems specified by `yn_periodic = 'n'` in `&system`, a 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 = 'n'` in `&system`, the unit system specified by this input keyword is used for most output files. To confirm the unit, see the first few lines of output files.

Options:

`'au'` or `'a.u.'` / atomic unit system

`'A_eV_fs'` / Angstrom-eV-fs unit system

4.4 &parallel

4.4.1 nproc_k

4.4.2 nproc_ob

4.4.3 nproc_rgrid(3)

integer, default=0

Options:

`nproc_k`/ Number of MPI parallelization for k-points of electron orbitals.

`nproc_ob`/ Number of MPI parallelization for orbital index of electron orbitals.

`nproc_rgrid(3)`'/ Number of MPI parallelization for each direction of real-space grid that are used for electron orbitals and density.

Defaults are `0` for `nproc_k`/`nproc_ob` and `(0,0,0)` for `nproc_rgrid`. In the default choice, MPI assignment is achieved atomatically. Users can specify `nproc_k`, `nproc_ob`, and `nproc_rgrid` manually. In that case, there are several constraints that should be fulfilled:

`nproc_k` must be set to `1` for `&system/yn_periodic='n'`.

`nproc_k` and `nproc_ob` must be set to `1` for `theory='maxwell'`.

`nproc_k` * `nproc_ob` * `nproc_rgrid(1)` * `nproc_rgrid(2)` * `nproc_rgrid(3)` = total number of processes.

4.4.4 yn_ffte

character, default='n'

Available for the DFT/TDDFT based options of `theory` with `&system/yn_periodic='y'`

For periodic systems, SALMON uses Fourier transformation to solve a poisson equation.

This switch selects if FFTE library is used or not. If FFTE is not used, the Fourier transformation in a simple algorithm is carried out.

Options

`'y'` / enable

`'n'` / disable

To enable it, following relations 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
```

4.4.5 yn_fftw

character, default='n'

Available for the DFT/TDDFT based options of `theory` with both `&system/yn_periodic='y'` and `&system/yn_periodic='n'`.

For isolated systems, this option is effective when `&poisson/method_poisson='ft'`

This switch selects if FFTW library is used or not. If FFTW is not used, the discrete Fourier transformation in a simple algorithm is carried out.

Caution: This variable is effective only when `--enable-fftw` is specified at the configure.

Options

`'y'` / enable

`'n'` / disable

4.4.6 yn_scalapack

character, default='n'

Available for `&calculation/theory='dft' or 'dft_md'`

To calculate large systems, an eigenvalue problem in the subspace diagonalization becomes a bottle-neck in the ground state calculation. In SALMON, ScaLAPACK library can be used to solve the eigenvalue problem.

To enable it, it is necessary to link ScaLAPACK library when you build SALMON.

Options:

`'y'` / enable

`'n'` / disable

4.4.7 yn_gramschmidt_blas

character, default='y'

Available for `&calculation/theory='dft' or 'dft_md'`

This switch selects if BLAS library is used or not in Gram Schmidt routines.

Options:

`'y'` / enable

`'n'` / disable

4.4.8 yn_eigenexa

character, default='n'

Available for `&calculation/theory='dft' or 'dft_md'`

SALMON can use RIKEN R-CCS EigenExa library to solve eigenvalue problem in subspace diagonalization. It is more efficient than ScaLAPACK to diagonalize matrices of large dimension. To enable it, it is necessary to link both ScaLAPACK and EigenExa libraries when you build SALMON.

Options:

'y' / enable
'n' / disable

4.4.9 yn_diagonalization_red_mem

character, Default='n'

Available for &parallel/yn_scalapack='y' or &parallel/yn_eigenexa='y'
This option reduces memory consumption in using ScaLAPACK/EigenExa libraries.
Options:
'y' / enable
'n' / disable

4.4.10 process_allocation

character, default='grid_sequential'

This controlls the order of process allocation.
Options:
'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'

4.5 &system

4.5.1 yn_periodic

character, default='n'

Available for all options of theory.
Specify boundary condition for electron orbitals.
Options:
'y' / periodic systems (crystalline solids)
'n' / isolated systems (molecules and nano-particles)

4.5.2 spin

character, default='unpolarized'

Available for the DFT/TDDFT based options of theory.
It specifies the spin state of the system, spin-unpolarized (closed shell) or spin-polarized (open shell).
Options
'unpolarized' / spin-unpolarized systems (default)
'polarized' / spin-polarized systems

'noncollinear' / noncollinear spin systems (see yn_spinorbit)

4.5.3 al(3)

real(8), default=0d0

Available for the DFT/TDDFT based options of theory.
Spatial box size or lattice constants for cuboid cell (x, y, z).
For nonorthogonal cell, see al_vec1(3), al_vec2(3), al_vec3(3).

4.5.4 al_vec1(3)

4.5.5 al_vec2(3)

4.5.6 al_vec3(3)

real(8), default=0d0

Available for yn_periodic = 'y' in the DFT/TDDFT based options of theory.
Primitive lattice vectors for nonorthogonal cell. For cuboid cell, see al(3).

4.5.7 nstate

integer, default=0

Available for the DFT/TDDFT based options of theory.
of orbitals/bands to be calculated. In the time evolution calculation of dielectrics, only occupied orbitals are evolved even when more nstate is specified.

4.5.8 nelec

integer, default=0

Available for the DFT/TDDFT based options of theory.
Number of valence electrons in the system.

4.5.9 nelec_spin(2)

integer, Default=0

Available for the DFT/TDDFT based options of theory.
Number of up/down-spin electrons are specified by nelec_spin(1)/nelec_spin(2).
This option is incompatible with nelec. (If nelec_spin is specified, nelec is ignored.)

4.5.10 temperature

real(8), default=-1d0

Available for DFT-based options of theory.

It specifies the temperature for electrons. The value must be given using the unit of energy as specified in `&units/unit_system`.

The kelvin unit can also be used by the keyword `temperature_k` instead of `temperature` (see next).

Occupation numbers are updated in every SCF step in the following way.

`temperature < 0` / the occupation numbers are fixed by `nelec` (appropriate for systems with energy gap).

`temperature = 0` / redistribution of the occupation numbers by the step function (metallic system at zero temperature).

`temperature > 0` / redistribution of the occupation numbers by the Fermi-Dirac distribution function.

4.5.11 temperature_k

real(8), default=-1d0

Available for DFT-based options of `theory`.

The same as `temperature` but kelvin is used as the unit.

4.5.12 nelem

integer, default=0

Available for the DFT/TDDFT based options of `theory`.

Number of atomic elements in the system.

4.5.13 natom

integer, default=0

Available for the DFT/TDDFT based options of `theory`.

Number of atoms in the system.

4.5.14 file_atom_red_coor

character, default='none'

Available for the DFT/TDDFT based options of `theory`.

Name of the file that contains atomic positions given in reduced coordinates. This option is incompatible with `&system/file_atom_coor`, `&atomic_coor`, and `&atomic_red_coor`.

4.5.15 file_atom_coor

character, default='none'

Available for the DFT/TDDFT based options of `theory`.

Name of the file that contains 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`.

4.5.16 yn_spinorbit

character, default='n'

Available for the DFT/TDDFT based options of `theory`.
Option for the spin-orbit coupling using the j-dependent pseudopotential formalism [Theurich & Hill, PRB 64, 073106 (2001)]. For pseudopotential(s), the UPF or VPS file format is required.

Options

`'y'` / enable (`spin='noncollinear'` is required. For `theory='dft'` mode, `method_mixing='simple'` or `method_mixing='simple_dm'` is recommended.)
`'n'` / disable (default)

4.5.17 yn_symmetry

[Trial] character, default='n'

Available for orthogonal cell system with the DFT/TDDFT based options of `theory`.
Symmetry option. Pre-generated input file, "sym.dat", is necessary. (details are not explained in the current manual)

Options

(e.g.) `'yyn'` / symmetry option is applied for the x and y direction (under applied electric field in the z-direction)
`'n'` / disable

4.5.18 absorbing_boundary

[Trial] character, default='none'

Available for the TDDFT based option of `theory` with orthogonal unit cell.
Absorbing boundary condition for electrons. (T. Nakatsukasa et al., J. Chem. Phys., 114, 2550 (2001))
Options:

`'none'` / disable (default)
`'z'` / absorbing boundary region is set in z direction for `'yn_periodic = 'y'`

4.5.19 imaginary_potential_w0

real(8), default='0d0'

Available when `absorbing_boundary` options is not `'none'`.
Strength of the absorbing (imaginary) potential.

4.5.20 imaginary_potential_dr

real(8), default='0d0'

Available when `absorbing_boundary` options is not `'none'`.

Thickness of the absorbing (imaginary) potential. For `absorbing_boundary='z'`, the absorbing region is 0 < z < `imagnary_potential_dr` and `al(3)-imagnary_potential_dr` < z < `al(3)`

4.6 &atomic_red_coor

Atomic coordinates in periodic systems (`'yn_periodoc = 'y'`) are specified in reduced coordinates using the 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, the first row for the first atom, etc. The number of rows must be equal to `&system/natom`. Atomic spicies are written in the first column although they are not used in the calculation. 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`.

4.7 &atomic_coor

Atomic coordinates are specified in the same way as `atomic_red_coor` but with length dimension. The unit chosen by `&units/unit_length` is applied.
This option is incompatible with `&system/file_atom_red_coor`, `&system/file_atom_coor`, and `&atomic_red_coor`.

4.8 &pseudo

4.8.1 izatom(:)

integer, default=-1

Available for the DFT/TDDFT based options of `theory`.
Atomic number of the element. The size of array is equal to `&system/nelem`.

4.8.2 file_pseudo(:)

character, default='none'

Available for the DFT/TDDFT based options of `theory`.
File name of the pseudopotential file. The size of array is equal to `&system/nelem`.

4.8.3 lmax_ps(:)

integer, default=-1

Available for the DFT/TDDFT based options of `theory`.
Maximum angular momentum of pseudopotential projectors.
If not given, values specified in the pseudopotential file will be used. The size of array is equal to `&system/nelem`.

4.8.4 lloc_ps(:)

integer, default=-1

Available for the DFT/TDDFT based options of `theory`.
Angular momentum of the pseudopotential that will be treated as local. The size of array is equal to `&system/nelem`.

4.8.5 yn_psmask(:)

[Trial] character, default='n'

Available for the DFT/TDDFT based options of `theory`.
Fourier filtering for pseudopotentials. The size of array is equal to `&system/nelem`.
Options:

`'y'` / enable
`'n'` / disable

4.8.6 alpha_mask(:)

[Trial] real(8), default=0.8d0

Available for the DFT/TDDFT based options of `theory`.
Parameter for the Fourier filtering of the pseudopotential. The size of array is equal to `&system/nelem`.

4.8.7 gamma_mask(:)

[Trial] real(8), default=1.8d0)

Available for the DFT/TDDFT based options of `theory`.
Parameter for the Fourier filtering of the pseudopotential. The size of array is equal to `&system/nelem`.

4.8.8 eta_mask(:)

[Trial] real(8), default=15.0d0

Available for the DFT/TDDFT based options of `theory`.
Parameter for the Fourier filtering of the pseudopotential. The size of array is equal to `&system/nelem`.

4.9 &functional

4.9.1 xc

character, default='none'

Available for the DFT/TDDFT based options of `theory`.

Exchange-correlation functional to be used.

In the present version, functionals 'PZ', 'PZM' and 'TBmBJ' are available for both `yn_periodic = 'y' and 'n'` calculations in the adiabatic approximation.

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). This potential is known to provide a reasonable description for the bandgap of various insulators. For this choice, the additional mixing parameter 'cval' may be specified. See below.

4.9.2 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 is evaluated by the formula in the original paper [Phys. Rev. Lett. 102, 226401 (2008)], $\langle|\nabla\rho(\mathbf{r};t)|/\rho(\mathbf{r};t)\rangle$. However, note that the value may be different from that in all electron calculations.

4.9.3 cname

4.9.4 xname

character, default='none'

Available for `theory='XXX'`.

XXX

4.9.5 alibxc

4.9.6 alibx

4.9.7 alibc

character, default='none'

Available for the DFT/TDDFT based options of `theory`.

Since version 1.1.0, exchange-correlation functionals in Libxc library (https://libxc.gitlab.io/) 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'
```

Note that, the hybrid functionals (hybrid gga/mgga) are not supported.

To use libxc libraries, `--enable-libxc` option must be added in excecuting configure. The available option of the exchange-correlation functionals are listed in the LibXC website. [See https://libxc.gitlab.io/functionals/]

4.10 &rgrid

4.10.1 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`.

4.10.2 num_rgrid(3)

integer, default=0

Available for the DFT/TDDFT based options of `theory`.
Number of real-space grids for each direction.
This cannot be used together with `&rgrid/dl`.

4.11 &kgrid

4.11.1 num_kgrid(3)

integer, default=1

Available for `yn_periodic='y'` in the DFT/TDDFT based options of `theory`.
Number of k-points (grid points of k-vector) for each direction discretizing the Brillouin zone.

4.11.2 dk_shift(3)

real(8), default=0d0

Available when `yn_periodic='y'` in the DFT/TDDFT based options of `theory`.
Shift of the k-vector. For the x-axis (or a*-axis for nonorthogonal cells), the k-vector is sampled as

k(1) = (dble(i1) - 0.5d0 + dk_shift(1)) / dble(num_kgrid(1)) - 0.5d0,
where i1 ranges from 1 to num_kgrid(1), and similarly for other directions.

4.11.3 file_kw

character, default='none'

Available for `yn_periodic='y'` in the DFT/TDDFT based options of `theory`.
File name for a file that includes user specified k-points. This file will be read if `num_kgrid` is equal to 0 or negative values. The file should be described in the following format :

```
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
```

4.12 &tgrid

4.12.1 nt

integer, Default=0

Available for 'dft_md' and TDDFT-based options of `theory`.
Number of total time steps for real-time propagation.

4.12.2 dt

real(8), Default=0d0

Available for 'dft_md' and TDDFT-based options of `theory`.
Time step size.

4.12.3 gram_schmidt_interval

integer, default=-1

Available for TDDFT-based options of `theory`.
Interval of a time step for the Gram-Schmidt orthonormalization of the orbitals during time evolution calculations. If this is set to a negative value, no Gram-Schmidt orthogonalization will be achieved. If this is set to zero, the Gram-Schumidt orthogonalization is carried out once at the initial step only. Usually this Gram-Schmidt orthogonalization is not necessary and should not be used.

4.13 &propagation

4.13.1 n_hamil

integer, default=4

Available for TDDFT-based options of `theory`.
Order of the Taylor expansion adopted for the propagation operator.

4.13.2 propagator

character, default=middlepoint

Available for TDDFT-based options of `theory`.
Choice of the propagator in the time evolution calculation.
Options:

`middlepoint` / Hamiltoinan at midpoint of two-times is used in the propagation if `yn_predictor_corrector = 'y'`. Hamiltoian at the time t is used if `yn_predictor_corrector = 'n'`.

`aetrs` / time-reversal symmetry propagator. [M.A.L. Marques, A. Castro, G.F. Bertsch, and A. Rubio, Comput. Phys. Commun., 151 60 (2003)].

4.13.3 yn_predictor_corrector

character, default='n'

Available for TDDFT-based options of `theory`.
Switch of the predictor-corrector method of TDDFT.
For meta-GGA functionals (`xc='tbmbj'`), the predictor corrector is automatically used even with `yn_predictor_corrector='n'`.
Options:

`'y'` / enable
`'n'` / disable

4.13.4 yn_fix_func

character, default='n'

Available for 'dft_md' and TDDFT-based options of `theory`.
Switch to fix the Hamiltonian during the time evolution, i.e., ground state Hamiltonian is used during the time propagation (frozen Hamiltonian calculation). With this option, the output file sysname_rt_energy.data contains only two columns. The excitation energy (the 2nd column) is calculated from the sum of the single-particle energies.
Options:

`'y'` / enable
`'n'` / disable

4.14 &scf

4.14.1 method_init_wf

character, default='gauss'

Available for 'dft' and 'dft_md' options of `theory`.
The generation method of the initial orbitals at the begening of the SCF iteration in DFT calculations. For a stable calculation of very large systems, multiple gaussian functions are preferred for a stable calculation.
Options:

`gauss` / single gauss function per orbital centered at a position determined by random numbers
`gauss2` / two gauss functions per orbital centered at positions determined by random numbers
`gauss3` / three gauss functions per orbital centered at positions determined by random numbers
`gauss4` / four gauss functions per orbital centered at positions determined by random numbers
`gauss5` / five gauss functions per orbital centered at positions determined by random numbers
`gauss10` / ten gauss functions per orbital centered at positions determined by random numbers
`random` / a random number is assigned at each real-space grid point of orbitals

4.14.2 method_init_density

[Trial] character, default='wf'

Available for 'dft' and 'dft_md' options of `theory`.
Specifying how to generate the initial density to start the SCF iteration in the DFT calculation.
Options:

`wf` / generate from the initial wavefunctions (cf. `method_init_wf`).
`pp` / generate from a superposition of the pseudo-atom densities. Supported for limited formats of pseudopotentials ('KY' and 'UPF').
`pp_magdir` / `pp` mode but with an initial spin polarization specified by `magdir_atom`. This option is for spin-polarized systems.
`read_dns_cube` / read the initial density from `dns.cube` file (Gaussian cube file format). The definition of r-grid & cell coordinate should be consistent.

4.14.3 magdir_atom

[Trial] real(8), default=0d0, 0d0, 0d0, ...

Available when `method_init_density=pp_magdir` is specified.
The array of initial values for the spin polarization at the respective atoms.

4.14.4 iseed_number_change

integer, default=0

Available for 'dft' and 'dft_md' options of `theory`.
Change a seed of random numbers that are used to generate initial orbitals. The value specified by this parameter is added to the seed.

4.14.5 nscf

integer, Default=300

Available for 'dft' and 'dft_md' options of `theory`.
Number of maximum SCF iterations in the DFT calculation.

4.14.6 method_min

character, Default='cg'

Available for 'dft' and 'dft_md' options of `theory`.
Method for updating orbitals in the SCF iteration. At present only confjugate gradient method is implemented.
Options:

`cg` / Conjugate-Gradient(CG) method

4.14.7 ncg

integer, default=4

Available for 'dft' and 'dft_md' options of `theory`.
Number of interations of conjugate-gradient method in the SCF iteration.

4.14.8 ncg_init

integer, default=4

Available for 'dft' and 'dft_md' options of `theory`.
Number of interations of conjugate-gradient method for the first SCF step.

4.14.9 method_mixing

character, default='broyden'

Available for 'dft' and 'dft_md' options of `theory`.
Method to update density/potential in the scf iteration.
Options:

`simple` / Simple mixing method for the density
`simple_potential` / Simple mixing method for the potential
`simple_dm` / Simple mixing method for the spin density matrix (available when `yn_spinorbit='y'`)
`broyden` / modified Broyden method for the density
`pulay` / Pulay method for the density

4.14.10 mixrate

real(8), default=0.5d0

Available for `method_mixing='simple'`, `method_mixing='simple_potential'`, or `method_mixing='simple_dm'` in 'dft' and 'dft_md' options of `theory`.

Mixing ratio for simple mixing.

4.14.11 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 the SCF iteration using the modified Broyden method. This must be less than 21.

4.14.12 alpha_mb

real(8), default=0.75d0

Available for `method_mixing='broyden'` in 'dft' and 'dft_md' options of `theory`.
A parameter of the modified Broyden method.

4.14.13 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 the SCF iteration using the Pulay method.

4.14.14 beta_p

real(8), default=0.75d0

Available for `method_mixing='pulay'` in 'dft' and 'dft_md' options of `theory`.
A parameter of the mixing rate of the Pulay method.

4.14.15 yn_auto_mixing

character, default='n'

Available for 'dft' and 'dft_md' options of `theory`.
Switch to change the mixing rate automatically (i.e. automatic adjustments of `mixrate`/`alpha_mb`/`beta_p`)
Options:

- `'y'` / enable
- `'n'` / disable

4.14.16 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`.

4.14.17 yn_subspace_diagonalization

character, default='y'

Available for 'dft' and 'dft_md' options of `theory`.
Switch for the subspace diagonalization during SCF iterations.
Options:
`'y'` / enable
`'n'` / disable

4.14.18 convergence

character, default='rho_dne'

Available for 'dft' and 'dft_md' options of `theory`.
Specify a quantity that is used for convergence check of the SCF iteration.
Options:
`'rho_dne'` / Convergence is checked by sum_ix|rho(ix,iter)-rho(ix,iter-1)|dx/N. N is `&system/nelec`.
`'norm_rho'` / Convergence is checked by the square of the norm of the density difference, $\|$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).

4.14.19 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 of convergence that is specified by `convergence` keyword.

4.14.20 nscf_init_redistribution

integer, default=10

Available for 'dft' and 'dft_md' options of `theory`.
Number of initial iterations during which a redistribution of the occupation number is suppressed in the finite temperature calculation.

4.14.21 nscf_init_no_diagonal

integer, default=10

Available for `&scf/yn_subspace_diagonalization='y'` in 'dft' option of `theory`.
Number of initial iterations during which the subspace diagonalization will not be carried out.

4.14.22 nscf_init_mix_zero

integer, default=-1

Available for 'dft' option of `theory`.

The density will not be mixed (i.e. fixed) during the given number of the SCF iteration, that is, orbitals are optimized without updating the density.

4.14.23 conv_gap_mix_zero

real(8), default=99999d0

Available if `nscf_init_mix_zero` is positive value in the 'dft' option of `theory`.

Specify a condition to quit the fixed density iteration forced by `step_initial_mix_zero` option. Mixing of the density will start after the band-gap energy exceeds this parameter for consecutive five SCF iteration steps.

4.14.24 yn_preconditioning

character, Default='n'

Available for `theory='dft'`.

Switch for preconditioning. The low-filter preconditioner is used when `&scf/yn_preconditioning='y'`.

Options:

`'y'` / enable

`'n'` / disable

4.14.25 alpha_pre

real(8), Default=0.6d0

Available for `theory='dft'`.

The certain factor used in preconditioning.

4.15 &emfield

4.15.1 trans_longi

character, default='tr'

Available for `yn_periodic='y'` in 'maxwell' and TDDFT based options of `theory`.

Specify the treatment of the polarization in the time evolution calculation.

Options:

`'tr'` / Transverse

`'lo'` / Longitudinal

`'2d'` / 2D maxwell-TDDFT (2D approximation) method (for more details, see `film_thickness` of &maxwell)

4.15.2 ae_shape1

4.15.3 ae_shape2

character, Default='none'

Available for 'maxwell' and TDDFT based options of `theory`.

Envelope shape of the first/second pulse. 'Acos2' indicates a cosine square envelope for vector potential, and 'Ecos2' a cosine square envelope for electric field.

Options:

`'impulse'` / A weak impulsive field is applied at $t = 0$. This will be used to explore linear response properties. The magnitude of the impulse can be specified by `e_impulse`.

`'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 an electric field.

`'Asin2cos'` [Trial] / Envelope of $\sin^2$with cosine type oscillation for a vector potential.

`'Asin2_cw'` [Trial] / Envelope of $\sin^2$at the beginning and continuous wave after that for a vector potential (for 'ae_shape1' only).

`'input'` [Trial] / read the vector potential as a numerical table with `file_input1` option (for 'ae_shape1' only).

`'none'` / no incident field is applied.

If 'Ecos2' is adopted, 'phi_cep1' must be chosen either 0.75 or 0.25, since otherwise the time integral of the electric field (vector potential at the end of the pulse) does not vanishi. There is no such restriction for 'Acos2' pulses.

For `yn_periodic='n'`, available choices are limited to `'impulse'`, `'Acos2'`, and `'Ecos2'`.

4.15.4 file_input1

character, default=''

Available if `ae_shape1='input'` is specified and `theory='tddft_pulse'`.

Name of an input file that contains user-defined vector potential. The file must be a numerical table separated by blank, having 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 using the units specified by `unit_system`. '#' and '!' may be used for a comment line.

Note that a linear interpolation will be applied when the time step differs from that used in the calculation.

4.15.5 e_impulse

real(8), Default=1d-2 a.u.

Available for 'maxwell' and TDDFT based options of `theory`.

Magnitude of the impulse in the impulsive perturbation. This valiable has the dimention of momentum, energy*time/length.

4.15.6 E_amplitude1

4.15.7 E_amplitude2

real(8), default=0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Maximum amplitude of electric field 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.

4.15.8 I_wcm2_1

4.15.9 I_wcm2_2

real(8), default=-1d0

Available for 'maxwell' and TDDFT based options of `theory`.
Maximum intensity (W/cm^2) of the first/second pulse. This valiable cannot be set with `&emfield/E_amplitude1` (`E_amplitude2`) simultaneously. For this quantity, a unit of W/cm^{2}is adopted irrespective of `&units\unit_system`.

4.15.10 tw1

4.15.11 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).
Note that this is not the FWHM duration.

4.15.12 omega1

4.15.13 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.

4.15.14 epdir_re1(3)

4.15.15 epdir_re2(3)

real(8), default=1d0, 0d0, 0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Real part of the polarization unit vector for the first/second pulse.

4.15.16 epdir_im1(3)

4.15.17 epdir_im2(3)

real(8), default=0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Imaginary part of the polarization unit vector for the first/second pulse. Using both real 'epdir_re1' and imaginary 'epdir_im1' parts of the polarization vector, circularly and general ellipsoidary polarized pulses may be described.

4.15.18 phi_cep1

4.15.19 phi_cep2

real(8), default=0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Carrier envelope phase of the first/second pulse. It specifies the CEP in unit of 2π.

4.15.20 t1_t2

real(8), default=0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Time-delay between the first and the second pulses.

4.15.21 t1_start

real(8), default=0d0

Available for 'maxwell' and TDDFT based options of `theory`.
Shift the starting time of the first pulse. (this is not available for multiscale option).

4.15.22 num_dipole_source

integer, default=0

Available for TDDFT based options of `theory`.
Number of radiation sources to mimic optical near fields. Maximum number is `2`.

4.15.23 vec_dipole_source(3,num_dipole_source)

real(8), default=0d0

Available for TDDFT based options of `theory`.
Dipole vectors of the radiation sources mimicing optical near fields.

4.15.24 cood_dipole_source(3,num_dipole_source)

real(8), default=0d0

Available for TDDFT based options of `theory`.
Coordinates of the radiation sources mimicing optical near fields.

4.15.25 rad_dipole_diele

real(8), default=2d0 [a.u.]

Available for TDDFT based options of `theory`.
Radii of dielectric spheres of the radiation sources mimicing optical near fields.

4.16 &singlescale[Trial]

4.16.1 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

4.16.2 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'.

4.16.3 yn_symmetrized_stencil

[Trial] character, default='n'

Available for `theory='single_scale_maxwell_tddft'`.
Switch to symmetrize the finite differences for the product of vector potential and orbitals, $(\nabla A(r) \cdot \psi(r))$. This option improves hermiticity of the Hamiltonian although computational cost increases.

4.16.4 yn_put_wall_z_boundary

[Trial] character, default='n'

Available for DFT/TDDFT based options of `theory`.

Option to put potential wall near the boundary planes at z=0 and z=‘‘&system/al(3)‘‘. This potential prevents electrons from crossing the z-boundary plane. In the single-scale Maxwell-TDDFT method, the electron density on the z-boundary plane harms the norm conservation of electrons due to the discontinuity of the vectorpotential. The wall is described using the square of cosine function.

Options:

`'y'` / put the potential wall

`'n'` / no potential wall

4.16.5 wall_height

real(8), default=100.0 [eV]

Available for `yn_put_wall_z_boundary='y'`.

The height of the potential wall.

4.16.6 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.

4.17 &multiscale

4.17.1 fdtddim

[Trial] character, default=’1d’

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` and `theory='maxwell_sbe'`

Dimension of macroscopic scale system (Maxwell(FDTD) calculation) in multiscale Maxwell-TDDFT method.

Options:

`'3d'` / 3-dimensional FDTD for macroscopic electromagnetism [currently not available]

`'1d'` / 1-dimensional FDTD (along the x-axis) for macroscopic electromagnetism

4.17.2 nx_m

integer, default=1

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` and `theory='maxwell_sbe'`

Number of macroscopic grid points inside materials for x-direction.

4.17.3 ny_m

integer, default=1

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` and `theory='maxwell_sbe'`
Number of macroscopic grid points inside materials for *y*-direction.

4.17.4 nz_m

[Trial] integer, default=1)

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` or `theory='maxwell_sbe'`
Number of macroscopic grid points inside materials for (*y*/*z*)-direction.

4.17.5 hx_m

real(8), default=0d0

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` or `theory='maxwell_sbe'`
Grid spacing of macroscopic coordinate for *x*-direction.
Variable `hx_m` is deprecated, and will be moved to `&units/dl_em(1)`

hy_m

4.17.6 hz_m

[Trial] real(8), default=0d0

Available for `theory='multi_scale_maxwell_tddft'` with `yn_periodic='y'` or `theory='maxwell_sbe'`
Grid spacing of macroscopic coordinate for (*y*/*z*)-direction.
Variable `hy_m` and `hz_m` are deprecated, and will be moved to `&units/dl_em(2:3)`

4.17.7 nxvacl_m

integer, default=1/0

Available for `theory='multi_scale_maxwell_tddft'` or `'maxwell_sbe'`
The parameter `nxvacl_m` will be replaced by `nxvac_m` and eventually removed in the future.

4.17.8 nxvacr_m

integer, default=1/0

Available for `theory='multi_scale_maxwell_tddft'` or `'maxwell_sbe'`
The parameter `nxvacr_m` will be replaced by `nxvac_m` and eventually removed in the future.

4.17.9 nxvac_m(2)

integer, default=0

Available for `theory='multi_scale_maxwell_tddft'` or `'maxwell_sbe'`

Represents the number of vacuum cells between the edge of the material region and the computational boundary. The first element of the array represents the number of cells from the leftmost cell (ix=1) on the x-axis to the left boundary. The second element represents the number of cells from the rightmost cell on the x-axis to the right boundary.

4.17.10 nyvac_m(2)

integer, default=0

Available for `theory='multi_scale_maxwell_tddft'` or `'maxwell_sbe'`
Provides same functionality of `nxvac_m(2)` for y-direction.

4.17.11 nzvac_m(2)

integer, default=0

Available for `theory='multi_scale_maxwell_tddft'` or `'maxwell_sbe'`
Provides same functionality of `nxvac_m(2)` for z-direction.

4.18 &maxwell

4.18.1 al_em(3)

real(8), default=0d0

Available for `theory='maxwell'`.
Size of simulation box in electromagnetic analysis.
Only two of `al_em`, `dl_em`, and `num_rgrid_em` must be set.

4.18.2 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.
Only two of `al_em`, `dl_em`, and `num_rgrid_em` must be set.

4.18.3 num_rgrid_em(3)

integer, default=0

Available for `theory='maxwell'`.
Number of real-space grids in electromagnetic analysis.
Only two of `al_em`, `dl_em`, and `num_rgrid_em` must be set.

4.18.4 at_em

real(8), default=0d0

Available for `theory='maxwell'`.
Total time for electromagnetic analysis.
Two of `at_em`, `dt_em`, and `nt_em` must be set.
Otherwise, both `at_em` and `nt_em` or either of those must be set.
(For the latter, `dt_em` is automatically determined from CFL condition)

4.18.5 dt_em

real(8), default=0d0

Available for `theory='maxwell'`.
Time step size for electromagnetic analysis.
If default is selected, this is automatically determined from CFL condition.

4.18.6 nt_em

integer, default=0

Available for `theory='maxwell'`.
Number of total time steps of time propagation in electromagnetic analysis.

4.18.7 boundary_em(3,2)

character, default='default'

Available for `theory='maxwell'` and `theory='multi_scale_maxwell_tddft'` and `theory='maxwell_sbe'`
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.
Options:
`'abc'` / absorbing boundary
`'pec'` / perfect electric conductor
`'periodic'` / periodic boundary

If `&system/yn_periodic='n'`, `'default'`, `'abc'`, and `'pec'` can be chosen, where `'default'` automatically chooses `'abc'`. If `&system/yn_periodic='y'`, `'default'`, `'abc'`, and `'periodic'` can be chosen, where `'default'` automatically chooses `'periodic'`.|
When `theory='maxwell'`, perfectly matched layer(PML) is used for `'abc'`.

4.18.8 shape_file

character, default='none'

Available for `theory='maxwell'`.
Name of input shape file in electromagnetic analysis. The shape file can be generated by using `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.9 media_num

integer, default=0

Available for `theory='maxwell'` and `theory='maxwell_sbe'`.
Number of media in electromagnetic analysis.

4.18.10 media_type(:)

character, default='vacuum'

Available for `theory='maxwell'` and `theory='maxwell_sbe'`
`media_type(n)` spesifies type of n-th media in electromagnetic analysis.
Options:

- `'vacuum'`
- `'constant media'`
- `'pec'`
- `'lorentz-drude'`

If `'lorentz-drude'` is chosen, linear response calculation is feasible by setting `&emfield/ae_shape1 or ae_shape2='impulse'`.
Besides, in the case of `theory='maxwell_sbe'`, `'multiscale'` is also available.

4.18.11 epsilon_em(:)

real(8), Default=1d0

Available for `theory='maxwell'`, `theory='maxwell_sbe'` and for TDDFT based options of `theory` with `trans_longi='2d'`.
For `theory='maxwell'`, `epsilon_em(n)` spesifies relative permittivity of n-th media in electromagnetic analysis.
For TDDFT based options of `theory` with `trans_longi='2d'`, the relative permittivity of the transparent media on both sides of the film is specified by `epsilon_em(1)` and `epsilon_em(2)`, respectively.

4.18.12 mu_em(:)

real(8), default=1d0

Available for `theory='maxwell'`.
`mu_em(n)` spesifies relative permeability of n-th media in electromagnetic analysis.

4.18.13 sigma_em(:)

real(8), default=0d0

Available for `theory='maxwell'`.
`sigma_em(n)` spesifies conductivity of n-th media in electromagnetic analysis.

4.18.14 pole_num_ld(:)

integer, default=1

Available for `theory='maxwell'`.
`pole_num_ld(n)` spesifies number of poles of n-th media, available for `type_media(n)='lorentz-drude'` in electromagnetic analysis.

4.18.15 omega_p_ld(:)

real(8), default=0d0

Available for `theory='maxwell'`.
`omega_p_ld(n)` spesifies plasma frequency of n-th media, available for `type_media(n)='lorentz-drude'` in electromagnetic analysis.

4.18.16 f_ld(:,:)

real(8), default=0d0

Available for `theory='maxwell'`.
`f_ld(n,m)` spesifies m-th oscillator strength of n-th media, available for `type_media='lorentz-drude'` in electromagnetic analysis. The first index is the media ID whose maximum value is given by `media_num`. The second index is the pole ID whose maximum value is given by `pole_num_ld(n)`.

4.18.17 gamma_ld(:,:)

real(8), default=0d0

Available for `theory='maxwell'`.
`gamma_ld(n,m)` spesifies m-th collision frequency of n-th media, available for `type_media(n)='lorentz-drude'` in electromagnetic analysis. The first index is the media ID whose maximum value is given by `media_num`. The second index is the pole ID whose maximum value is given by `pole_num_ld(n)`.

4.18.18 omega_ld(:,:)

real(8), default=0d0

Available for `theory='maxwell'`.
`omega_ld(n,m)` spesifies m-th oscillator frequency of n-th media, available for `type_media(n)='lorentz-drude'` in electromagnetic analysis. The first index is the media ID whose maximum value is given by `media_num`. The second index is the pole ID whose maximum value is given by `pole_num_ld(n)`.

4.18.19 wave_input

character, default='none'

Available for `theory='maxwell'`.
If `'source'`, the incident pulse in electromagnetic analysis is generated by the incident current source.

4.18.20 ek_dir1(3)

4.18.21 ek_dir2(3)

real(8), default=0d0

Available for `theory='maxwell'`.
Propagation direction of the first/second pulse (*x*, *y*, and *z* directions). Each component must be 0d0 or 1d0.

4.18.22 source_loc1(3)

4.18.23 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'`.

4.18.24 gbeam_sigma_plane1(3)

4.18.25 gbeam_sigma_plane2(3)

4.18.26 gbeam_sigma_line1(3)

4.18.27 gbeam_sigma_line2(3)

[Trial] real(8), default=-1d0

Available for `theory='maxwell'` with `wave_input='source'`.
These input keywords specify the width of Gauss function, exp(-0.5(abs(r-r_0)/sigma)^2), applied for the incident current source to generate the first/second pulse. These input keywords work only when their values > 0.0d0. The center of the Gauss function, r_0, is specified by `source_loc1/2`.
`gbeam_sigma_plane1/2` specifies the width of 2D Gauss function (*xy*, *yz*, and *xz* planes).
`gbeam_sigma_line1/2` specifies the width of 1D Gauss function (*x*, *y*, and *z* axes).

4.18.28 obs_num_em

integer, default=0

Available for `theory='maxwell'`.
Number of observation points in electromagnetic analysis. From the obtained results, figure and animation files can be generated by using SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.29 obs_samp_em

integer, default=0

Available for `theory='maxwell'`.

Sampling time-step of the observation in electromagnetic analysis.
If default is selected, this is automatically determined.

4.18.30 obs_loc_em(:,3)

real(8), default=0d0

Available for `theory='maxwell'`.
`obs_loc_em(n,1:3)=x,y,z` spesifies location of the n-th 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'`.

4.18.31 obs_plane_ene_em(:,:)

real(8), default=-1d0

Available for `theory='maxwell'`.
`obs_loc_em(n,:)=energy1,energy2,energy3,...` spesifies energy value of the n-th observation point in electromagnetic analysis. At the spesified energies, Fourier-transformed spatial distributions on the xy, yz, and xz plans are outputed. This input keyword must be larger than 0.

4.18.32 yn_obs_plane_em(:)

character, default='n'

Available for `theory='maxwell'`.
Spesify whether or not to generate output of the electrmagnetic fields on the planes (*xy*, *yz*, and *xz* planes) for n-th observation point. This option must be `'y'` for generating animation files by using `FDTD_make_figani` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
Options:

`'y'`
`'n'`

4.18.33 yn_obs_plane_integral_em(:)

character, default='n'

Available for `theory='maxwell'`.
Specify whether or not to generate output of the spatial integration of electromagnetic fields on the planes (*xy*, *yz*, and *xz* planes) for n-th observation point.
Options:

`'y'`
`'n'`

4.18.34 yn_wf_em

character, default='y'

Available for `theory='maxwell'`.

Switch of a window function for linear response calculation.
Options:

'y'
'n'

4.18.35 film_thickness

real(8), default=0d0

Available for TDDFT based options of `theory` with `trans_longi='2d'`.
Thickness of the film for the 2D maxwell-TDDFT (2D approximation) method [S. Yamada and K. Yabana, PRB 103, 155426 (2021)].
For a slab system, `film_thickness` should be set to the side length of the calculation cell, i.e., the thickness of the slab plus the length of the vacuum region [S. Yamada et al., PRB 98, 245147 (2018)].
The relative permittivity of the transparent media on both sides of the film can be specified by `epsilon_em(1)` and `epsilon_em(2)`, respectively (default=vacuum).

4.18.36 media_id_pml(3:2)

integer, default=0

Available for `theory='maxwell'`.
Media ID used in PML. 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.

4.18.37 media_id_source1

4.18.38 media_id_source2

integer, default=0

Available for `theory='maxwell'`.
Media ID used in incident current source1/source2 to generate the first/second pulse.

4.18.39 bloch_k_em(3)

[Trial] real(8), default=0d0

Available for `theory='maxwell'` with `yn_periodic='y'`.
Wavenumber used in Bloch boundary conditions. When sum(**|bloch_k_em(:)|**)>0, Bloch boundary conditions are automatically applied.

4.18.40 bloch_real_imag_em(3)

[Trial] character, default='real'

Available for `theory='maxwell'` with `yn_periodic='y'` and sum(**|bloch_k_em(:)|**)>0.
Specify real or imaginary parts for exp(ikr) used in Bloch boundary conditions.
Options:

'real'

'imag'

4.18.41 ase_num_em

integer, default=0

Available for theory='maxwell' with yn_periodic='n'.
Number of energy or wavelength grid points specified by ase_ene_min_em/ase_ene_max_em or ase_wav_min_em/ase_wav_max_em.
If this is specified as larger than 0, Absorption-, Scattering-, and Extinction-cross-sections will be outputed at the end of calculation.
Those are normalized by the spectral distribution of the incident pulse.

4.18.42 ase_ene_min_em

4.18.43 ase_ene_max_em

real(8), default=-1d0

Available for theory='maxwell' with ase_num_em>0 and yn_periodic='n'.
Energy range for Absorption-, Scattering-, and Extinction-cross-sections.

4.18.44 ase_wav_min_em

4.18.45 ase_wav_max_em

real(8), default=-1d0

Available for theory='maxwell' with ase_num_em>0 and yn_periodic='n'.
Wavelength range for Absorption-, Scattering-, and Extinction-cross-sections.

4.18.46 ase_smedia_id_em

integer, default=0

Available for theory='maxwell' with ase_num_em>0 and yn_periodic='n'.
Media ID used as surrounding media.

4.18.47 ase_box_cent_em(3)

real(8), default=0d0

Available for theory='maxwell' with ase_num_em>0 and yn_periodic='n'.
ase_box_cent_em(1:3)=x,y,z spesifies location of the center of a closed surface (box shape) to calculate Absorption-, Scattering-, and Extinction-cross-sections.

4.18.48 ase_box_size_em(3)

real(8), default=-1d0

Available for `theory='maxwell'` with `ase_num_em>0` and `yn_periodic='n'`.
`ase_box_size_em(1:3)=X,Y,Z` spesifies size of a closed surface (box shape) to calculate Absorption-, Scattering-, and Extinction-cross-sections.

4.18.49 art_num_em

integer, default=0

Available for `theory='maxwell'` with `yn_periodic='y'`.
Number of energy or wavelength grid points specified by `art_ene_min_em/art_ene_max_em` or `art_wav_min_em/art_wav_max_em`.
If this is specified as larger than 0, Absorption-, Reflection-, and Transmission-ratas will be outputed at the end of calculation.
Those are normalized by the spectral distribution of the incident pulse.

4.18.50 art_ene_min_em

4.18.51 art_ene_max_em

real(8), default=-1d0

Available for `theory='maxwell'` with `art_num_em>0` and `yn_periodic='y'`.
Energy range for Absorption-, Reflection-, and Transmission-ratas.

4.18.52 art_wav_min_em

4.18.53 art_wav_max_em

real(8), default=-1d0

Available for `theory='maxwell'` with `art_num_em>0` and `yn_periodic='y'`.
Wavelength range for Absorption-, Reflection-, and Transmission-ratas.

4.18.54 art_smedia_id_em

integer, default=0

Available for `theory='maxwell'` with `art_num_em>0` and `yn_periodic='y'`.
Media ID used as surrounding media.

4.18.55 art_plane_bot_em(3)

4.18.56 art_plane_top_em(3)

real(8), default=0d0

Available for `theory='maxwell'` with `art_num_em>0` and `yn_periodic='y'`.

`art_plane_bot_em(1:3)=x1,y1,z1` and `art_plane_top_em(1:3)=x2,y2,z2` spesify location of bottom and top planes on the propagation axis to calculate Absorption-, Reflection-, and Transmission-ratas.

4.18.57 yn_make_shape

character, default='n'

Available for `theory='maxwell'`.
Switch for making shape. This is same functionality for `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
Options:

`'y'`
`'n'`

4.18.58 yn_output_shape

character, default='n'

Available for `theory='maxwell'`.
Switch for outputing shape file in cube format when `yn_make_shape='y'`.
Options:

`'y'`
`'n'`

4.18.59 yn_copy_x

4.18.60 yn_copy_y

4.18.61 yn_copy_z

character, default='n'

Available for `theory='maxwell'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
Options:

`'y'`
`'n'`

4.18.62 rot_type

character, default='radian'

Available for `theory='maxwell'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).
Options:

`'radian'`
`'degree'`

4.18.63 n_s

integer, default=0

Available for `theory='maxwell'` and `theory='maxwell_sbe'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.64 typ_s(:)

character, default='none'

Available for `theory='maxwell'` and `theory='maxwell_sbe'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.65 id_s(:)

integer, default=0

Available for `theory='maxwell'` and `theory='maxwell_sbe'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.66 inf_s(:,10)

real(8), default=0

Available for `theory='maxwell'` and `theory='maxwell_sbe'`.
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.18.67 ori_s(:,3)

4.18.68 rot_s(:,3)

real(8), default=0d0

Available for `theory='maxwell'`
See `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html).

4.19 &analysis

4.19.1 projection_option

character, default='no'

Available for TDDFT based options of `theory`.
Methods of projection to analyze the excited states (e.g. the number of excited electrons).
Output files: SYSname_nex.data, SYSname_ovlp.data
Options:

`'no'` / no projection.

`'gs'` / projection to eigenstates of the ground-state Hamiltonian whose k-point is shifted as k+A(t)/c (i.e. Houston functions).

'td' / projection to instantaneous eigenstates of the time-dependent Hamiltonian.

4.19.2 out_projection_step

integer, default=100

Available when projection_option is specified.
Resuts of the projection analysis will be outputted every out_projection_step steps during the time-propagation.

4.19.3 threshold_projection

real(8), default=1e-6

Available when projection_option is specified.
Convergence threshold for the iteration of the eigenstates calculation.z

4.19.4 yn_out_intraband_current

character, default='n'

Available when projection_option is specified.
Switch for output of the intra-band current density [T. Otobe, Phys. Rev. B 94, 235152 (2016).].
Output file: SYSname_intra_current.data
Options:

'y' / enable
'n' / disable

4.19.5 nenergy

integer, default=1000

Number of energy grid points for frequency-domain analysis. This parameter is used, for examples, in theory='tddft_response' and theory='maxwell'.

4.19.6 de

real(8), Default=0.01d0 (eV)

Energy grid size for frequency-domain analysis.
This parameter is used, for examples, in theory='tddft_response' and theory='maxwell'.

4.19.7 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.

4.19.8 yn_out_rt_energy_components

character, default='n'

Available for the TDDFT based option of `theory`.
Switch for printing out the energy components such as the kinetic energy term. The total energy is the sum of the energy components.
Options:

`'y'` / enable
`'n'` / disable

4.19.9 yn_out_psi

character, default='n'

Available for `theory='dft'`.
Switch for output of orbitals.
Options:

`'y'` / enable
`'n'` / disable

The format of the output is specified by &analysis/`format_voxel_data`.

4.19.10 yn_out_dos

character, default='n'

Available for `theory='dft'`.
Switch for output of density of states.
Options:

`'y'` / enable
`'n'` / disable

4.19.11 yn_out_pdos

character, default='n'

Available for `theory='dft'`.
Switch for output of projected density of states.
Options:

`'y'` / enable
`'n'` / disable

4.19.12 yn_out_dos_set_fe_origin

character, default='n'

Available when `yn_out_dos='y'` or `yn_out_pdos='y'`.
Switch to set the Fermi energy to zero in the energy axis of DoS.
If the temperature is not specified, this option sets the valence band maximum to zero.
Options:

'y' / enable
'n' / disable

4.19.13 out_dos_start

real(8), default=-1d10 (eV)

4.19.14 out_dos_end

real(8), default=1d10 (eV)

Available when yn_out_dos='y' or yn_out_pdos='y'.
Lower/Upper bound of the energy range for the density of states 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.

4.19.15 out_dos_nenergy

integer, default=601

Available when yn_out_dos='y' or yn_out_pdos='y'.
Number of energy points sampled in the density of states spectra.

4.19.16 out_dos_function

character, default='gaussian'

Available when yn_out_dos='y' or yn_out_pdos='y'.
Choice of the smearing function for the density of states spectra.
Options:

gaussian / Gaussian function
lorentzian / Lorentzian function

4.19.17 out_dos_width

real(8), default=0.1d0 [eV]

Available when yn_out_dos='y' or yn_out_pdos='y'.
Smearing width used in the density of states spectra.

4.19.18 yn_out_dns

character, default='n'

Available for theory='dft'.
Switch to output electron density distribution of the ground state.
Options:

'y' / enable
'n' / disable

4.19.19 yn_out_dns_rt

character, default='n'

Available when `theory='dft_md' or 'theory=tddft_pulse'`.
Switch to output electron density distribution during the time-propagation.
Options:

`'y'` / enable
`'n'` / disable

4.19.20 out_dns_rt_step

integer, default=50

Available when `theory='dft_md' or 'theory=tddft_pulse'`.
Density is outputted every `out_dns_rt_step` steps.

4.19.21 yn_out_dns_ac_je

character, default='n'

Available for `theory='single_scale_maxwell_tddft'`.
Switch to print the electron density, vector potential, electronic current, and ionic coordinates every `out_dns_ac_je_step` time steps.
Options:

`'y'` / enable
`'n'` / disable

The data written in binary format are divided into files corresponding to the space-grid parallelization number.

4.19.22 out_dns_ac_je_step

integer, default=50

Available for `theory='single_scale_maxwell_tddft'`.
Electron density, vector potential, electronic current, and ionic coordinates are outputted every `outdns_dns_ac_je_step` time steps.

4.19.23 yn_out_micro_je

character, default='n'

Available for TDDFT based methods.
Switch to print the microscopic electron current density (`je_micro_***` files) at every `out_micro_je_step` time steps.
Options:

`'y'` / enable
`'n'` / disable

4.19.24 out_micro_je_step

integer, default=50

Available for TDDFT based methods with `yn_out_micro_je='y'`.
See `yn_out_micro_je`.

4.19.25 yn_out_dns_trans

[currently not available] character default='n'

Available for `theory='tddft_pulse'`.
Switch to calculate transition density at specified frequency omega (specified by `out_dns_trans_energy`), drho(r,omega)=FT(rho(r,t)-rho_gs(r))/T.
Options:

- `'y'` / enable
- `'n'` / disable

4.19.26 out_dns_trans_energy

[currently not available] real(8), default=1.55d0 [eV]

Available for `theory='tddft_pulse'`.
A frequency to output drho(r,omega)=FT(rho(r,t)-rho_gs(r))/T.

4.19.27 yn_out_elf

character, default='n'

Available for `theory='dft'`.
Switch to output the electron localization function.
Options:

- `'y'` / enable
- `'n'` / disable

4.19.28 yn_out_elf_rt

character, default='n'

Available for `theory='dft_md', 'tddft_pulse'`.
Switch to output the electron localization function during the time propagation every `out_elf_rt_step` time steps.
Options:

- `'y'` / enable
- `'n'` / disable

4.19.29 out_elf_rt_step

integer, default=50

Available for `theory='dft_md', 'tddft_pulse'`.
Electron localization function during the time propagation is outputted every `out_elf_rt_step` time steps.

4.19.30 yn_out_estatic_rt

character, default='n'

Available for `theory='tddft_pulse'`.
Switch to print the static electric field during the time propagation every `out_estatic_rt_step` time steps.
Options:

`'y'` / enable
`'n'` / disable

4.19.31 out_estatic_rt_step

integer, default=50

Available for `theory='tddft_pulse'`.
The static electric field during the time propagation is outputed every `out_estatic_rt_step` time steps.

4.19.32 yn_out_rvf_rt

character, default='n'

Available for TDDFT based options and 'dft_md' option of `theory`.
Switch to print the coordinates[A], velocities[au], forces[au] of atoms during time-propagation in `SYSname`_trj.xyz every `out_rvf_rt_step` time steps.
Options:

`'y'` / enable
`'n'` / disable

If `yn_md='y'`, this option is automatically turned on.

4.19.33 out_rvf_rt_step

integer, default=10

Available for TDDFT based options and 'dft_md' option of `theory`.
The coordinates[A], velocities[au], forces[au] of atoms during time-propagation are outputed in `SYSname`_trj.xyz every `out_rvf_rt_step` time steps.

4.19.34 yn_out_tm

[Trial] character, default='n'

Available for `yn_periodic='y'` with `theory='dft'`.
Switch to calculate and print the transition matrix elements between occupied and virtual orbitals to `SYSname`_tm.data after the ground state calculation.

Options:

'y' / enable

'n' / disable

4.19.35 yn_out_gs_sgm_eps

[Trial] character, default='n'

Available for theory='dft'.

Switch to calculate and print conductivity (sigma) and dielectric function (epsilon) based on transition moment after convergence of the ground state calculation. These are printed in the output files, SYSname_sigma.data and SYSname_epsilon.data

'y' / enable

'n' / disable

4.19.36 out_gs_sgm_eps_mu_nu

integer, default=3,3

Available for yn_out_gs_sgm_eps='y' with theory='dft'.

Index of conductibity and dielectric tensol element calculated in this option. Default of (3,3) means zz element.

4.19.37 out_gs_sgm_eps_width

real(8), default=0.015d0 [eV]

Available for yn_out_gs_sgm_eps='y' with theory='dft'.

Smearing width used in conductivity and dielectric function

4.19.38 out_ms_step

integer, default=100

Available for theory='multi_scale_maxwell_tddft'.

Some quantities are printed every out_ms_step time step in the Maxwell-TDDFT multiscale calculations.

4.19.39 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

4.19.40 nsplit_voxel_data

integer, default=1

Available for `format_voxel_data='avs'`.
Number of separated files for three dimensional data.

4.19.41 yn_lr_w0_correction

[Trial] character, default='n'

Available for `yn_periodic='y'` and `trans_longi='tr'` with `theory='tddft_response'`.
Apply correction around zero frequency of dielectric function to suppress numerical error.
Options:

- `'y'` / enable
- `'n'` / disable

4.19.42 yn_out_current_decomposed

character, default='n'

Available for TDDFT based options of `theory`.
Switch to output docomposed elements of the electron current density.
The sum of the docomposed elements is equal to the current density in SYSname_rt.data.
Output file: SYSname_current_decomposed.data
Options:

- `'y'` / enable
- `'n'` / disable

4.19.43 out_current_decomposed_step

integer, default=100

Available when `yn_out_current_decomposed='y'`.
The decomposed current data is outputted every `out_current_decomposed_step` step.

4.19.44 out_rt_spin_step

integer, default=100

Available for TDDFT based methods with `spin='noncollinear'`.
The spin magnetization and spin current density are outputted every `out_rt_spin_step` time steps in the output file SYSname_rt_spin.data.
For the definition of the spin current, see [N. Tancogne-Dejean et al, npj Computational Materials 8, 145 (2022).].

4.19.45 yn_out_mag_decomposed_rt

character, default='n'

Available for TDDFT based methods with `spin='noncollinear'`.
Switch to output docomposed elements of the time-dependent spin magnetization at every `out_rt_spin_step` time steps.
Output file: SYSname_mag_decomposed_rt.data
Options:

`'y'` / enable
`'n'` / disable

4.19.46 yn_out_mag_micro_rt

character, default='n'

Available for TDDFT based methods with `spin='noncollinear'`.
Switch to output voxel data files of the microscopic magnetization at every `out_rt_spin_step` time steps.
Output files: mag_micro_[xyz]_000001.<format_voxel_data> etc.
Options:

`'y'` / enable
`'n'` / disable

4.19.47 yn_out_spin_current_decomposed

character, default='n'

Available for TDDFT based methods with `spin='noncollinear'`.
Switch to output docomposed elements of the spin current density at every `out_rt_spin_step` time steps.
Output file: SYSname_spin_current_decomposed.data
Options:

`'y'` / enable
`'n'` / disable

4.19.48 yn_out_spin_current_micro

character, default='n'

Available for TDDFT based methods with `spin='noncollinear'`.
Switch to output voxel data files of the microscopic spin-current density at every `out_rt_spin_step` time steps.
Output files: spin_curr_micro_[xyz]_[xyz]_000001.<format_voxel_data> etc.
Options:

`'y'` / enable
`'n'` / disable

4.19.49 yn_out_perflog

character, default='y'

Available for all `theory`
Switch to print the performance log of routines and modules.
Options:
`'y'` / enable
`'n'` / disable

4.19.50 format_perflog

character, default='stdout'

Available for `yn_out_perflog = 'y'`
The output format of performance log.
Options:
`'stdout'` / standard output unit
`'text'` / save as a text file
`'csv'` / save as a csv format file

4.20 &poisson

4.20.1 method_poisson

character, Default='cg'

Available for `yn_periodic='n'` in DFT and TDDFT based options of `theory`.
This papameter specify how to solve the Poisson equation.
Options:
`cg`/ Conjugate-Gradient(CG) method
`ft`/ Fourier transformation method
`dirichlet`/ Dirichlet boundary condition method

4.20.2 layout_multipole

character, Default=3

Available for `yn_periodic='n'` in DFT and TDDFT based options of `theory`.
This papameter specify how to achieve multipole expansioin in the Hartree potential calculation.
Options:
`1`/ A single pole at the center.
`2`/ Multipoles are set at each center of atoms.
`3`/ Multipoles are set at the center of mass of electrons in prepared cuboids in each process.

4.20.3 num_multipole_xyz(3)

integer, default=0

Available for `yn_periodic='n'` in DFT and TDDFT based options of `theory`.
Number of multipoles. When default is set, the number of multipoles is calculated automatically.

4.20.4 lmax_multipole

[Trial] integer, default=4

Available for `yn_periodic='n'` in DFT and TDDFT based options of `theory`.
A maximum order of the multipole expansion to prepare boundary condition of Poisson equation.

4.20.5 threshold_cg

real(8), default=1d-15 [a.u.]

Available for `yn_periodic='n'` in DFT and TDDFT based options of `theory`.
A threshold for the convergence of the Hartree-cg calculation. A quantity examined is given by ||tVh(i)-tVh(i-1)||^2/(number of grids).

4.21 &ewald

4.21.1 newald

integer, default=4

Available for `yn_periodic='y'` in DFT/TDDFT based options of `theory`.
Parameter of the Ewald method for the ion-ion Coulombic interaction. Short-range part of the Ewald sum is calculated within `newald`-th nearlist neighbor cells.

4.21.2 aewald

real(8), default=0.5d0 [a.u.]

Available for `yn_periodic='y'` in DFT/TDDFT based options of `theory`.
Square of range separation parameter for Ewald method (This parameter is given only in atomic unit).

4.21.3 cutoff_r

real(8), default=-1d0

Available for `yn_periodic='y'` in DFT/TDDFT based options of `theory`.
Cut-off length in real-space. The length is automatically determined if `cutoff_r` < 0.

4.21.4 cutoff_r_buff

real(8), default=2d0 [a.u.]

Available for `yn_periodic='y'` in `yn_md='y'` or in `theory='dft_md'`.
Buffer length in radius for book-keeping for real-space interaction.

4.21.5 cutoff_g

real(8), Default=-1d0

Available for `yn_periodic='y'` in DFT/TDDFT based options of `thddeory`.
Cut-off in G-space in the Ewald method. No cut-off in default.

4.22 &opt[Trial]

4.22.1 nopt

integer, default=100

Available for `yn_opt='y'` in `theory='dft'`.
The maximum step number of geometry optimization.

4.22.2 convrg_opt_fmax

real(8), default=1d-3 (a.u.)

Available for `yn_opt='y'` in `theory='dft'`.
Convergence threshold of geometry optimization is specified for the maximum force acting on atoms.

4.22.3 max_step_len_adjust

real(8), default=-1d0

Available for `yn_opt='y'` in `theory='dft'`.
Set maximum optimization step length (if positive number is given)

4.23 &md[Trial]

4.23.1 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)

4.23.2 thermostat

character, default='nose-hoover'

Available for `yn_md='y'` or `theory='dft_md'`.
Thermostat in "NVT" option:
Options:

`nose-hoover`/ Nose-Hoover thermostat

4.23.3 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.

4.23.4 step_update_ps

integer, default=10

Available for `yn_md='y'` or `theory='dft_md'`.
Time step interval for updating pseudopotential (Larger number reduces computational time but increases inaccuracy).

4.23.5 temperature0_ion_k

real(8), Default=298.15d0 [K]

Available for `yn_md='y'` or `theory='dft_md'`.
Setting ionic temperature in unit of [K] for NVT ensemble, velocity scaling and generating initial velocities.

4.23.6 yn_set_ini_velocity

character, Default='n'

Available for `yn_md='y'` or `theory='dft_md'`.
Switch to generate initial velocities.
Options:

`y`/ Generate initial velocity with Maxwell-Bortzman distribution
`n`/ disable

4.23.7 file_ini_velocity

[Trial] character, default='none'

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).

4.23.8 thermostat_tau

real(8), default=1d0 [fs]

Available for `yn_md='y'` or `theory='dft_md'`.
Parameter in Nose-Hoover method: controlling time constant for temperature.

4.23.9 yn_stop_system_mom

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

4.24 &jellium

4.24.1 yn_jm

character, default='n'

Available for the DFT/TDDFT based options of `theory`.
Switch to use jellium model.
Options:
- `y`/ enable
- `n`/ disable

When `yn_jm='y'`, `&functional/xc` must be `'pz'`.

4.24.2 yn_charge_neutral_jm

character, default='y'

Available for `yn_jm='y'` in the DFT/TDDFT based options of `theory`.
Option to enforce exact charge neutrality :
Options:
- `y`/ enable. `rs_bohr_jm` is modified to fulfill exact charge neutrality.
- `n`/ disable. `rs_bohr_jm` is not modified, and there may appears small charge-neutrality error.

4.24.3 yn_output_dns_jm

character, default='y'

Available for `yn_jm='y'` in the DFT/TDDFT based options of `theory`.
Switch to output positive background charge density.
Options:
- `y`/ enable
- `n`/ disable

4.24.4 shape_file_jm

character, default='none'

Available for `yn_jm='y'` in the DFT/TDDFT based options of `theory`.
Name of input shape file that contains positive background charge density to be used in the jellium model calculations. The shape file can be generated by using `FDTD_make_shape` in SALMON utilities (https://salmon-tddft.jp/utilities.html). When `shape_file_jm='none'`, the shape of the positive background charge density is specified by `sphere_nion_jm` and `sphere_loc_jm` which generate spherical shapes.

4.24.5 num_jm

integer, Default=0

Available for `yn_jm='y'` in the DFT/TDDFT based options of `theory`.
When `shape_file_jm` is not 'none', `num_jm` specifies number of media used in the jellium model. When `shape_file_jm='none'`, `num_jm` specifies number of spherical shapes.

4.24.6 rs_bohr_jm(:)

real(8), default=0d0

Available for `yn_jm='y'` in the DFT/TDDFT based options of `theory`.
When `shape_file_jm` is not 'none', `rs_bohr_jm(n)` spesifies the Wigner-Seitz radius of n-th media. When `shape_file_jm='none'`, `rs_bohr_jm(n)` spesifies the Wigner-Seitz radius of n-th sphere.

4.24.7 sphere_nion_jm(:)

integer, default=0

Available for `yn_jm='y'` and `shape_file_jm='none'` in the DFT/TDDFT based options of `theory`. `sphere_nion_jm(n)` spesifies ion number for n-th sphere. At present, only neutral systems can be treated.

4.24.8 sphere_loc_jm(:,3)

real(8), default=0d0

Available for `yn_jm='y'` and `shape_file_jm='none'` in the DFT/TDDFT based options of `theory`. `sphere_loc_jm(n,1:3)=x,y,z` spesifies location of center of mass for n-th sphere. Note that the coordinate system ranges from `-al/2` to `al/2` for `&system/yn_periodic='n'` while ranges from `0` to `al` for `&system/yn_periodic='y'`.

4.25 &code

4.25.1 yn_want_stencil_hand_vectorization

character, default='y'

Switch to use hand-vectorized optimization code of stencil in the hamiltonian calculation.
SALMON checks if the calculation can use the hand-vectorized code. If it fails, SALMON will use a typical implementation.

4.25.2 yn_want_communication_overlapping

character, default='n'

Available for `theory='tddft*' or '*maxwell_tddft'`
Switch to use computation/communication overlap algorithm to improve the performance of stencil in the hamiltonian calculation.
SALMON checks if the calculation can use the overlap algorithm. If it fails, SALMON will uses a non-overlap algorithm.

4.25.3 stencil_openmp_mode

character, default='auto'

This option selects an OpenMP parallelization mode of stencil in the hamiltonian calculation.
Options:

`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.

4.25.4 current_openmp_mode

character, default='auto'

This selects an OpenMP parallelization mode of the current calculation.
Options:

`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.

4.25.5 force_openmp_mode

character, default='auto'

This selects an OpenMP parallelization mode of the force calculation.
Options:

`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.

4.26 &sbe

4.26.1 num_sbe

integer, default=1

Number of materials in the Maxwell-SBE calculations.

4.26.2 sysname_sbe(:)

character, default='default'

System name of each material in the Maxwell-SBE calculations. The index is the media ID whose maximum value is given by `num_sbe`.

4.26.3 nk_sbe(:)

integer, default=-1

Total number of k-points in each material in the Maxwell-SBE calculations. The index is the media ID whose maximum value is given by `num_sbe`.

4.26.4 nstate_sbe(:)

integer, default=-1

orbitals/bands to be calculated in each material in the Maxwell-SBE calculations. The index is the media ID whose maximum value is given by `num_sbe`.

4.26.5 nelec_sbe(:)

integer, default=-1

Number of valence electrons in each material in the Maxwell-SBE calculations. The index is the media ID whose maximum value is given by `num_sbe`.

4.26.6 al_sbe(:,:)

real(8), default=0.d0

Spatial box size or lattice constants for cuboid cell (x, y, z) in each material in the Maxwell-SBE calculations. The first index(1-3 rows) corresponds to x, y, and z axes. The second index is the media ID whose maximum value is given by `num_sbe`.

4.26.7 al_vec_1_sbe(:,:)

4.26.8 al_vec_2_sbe(:,:)

4.26.9 al_vec_3_sbe(:,:)

integer, default=-1

Primitive lattice vectors for nonorthogonal cell in each material in the Maxwell-SBE calculations. The first index(1-3 rows) corresponds to x, y, and z components of the lattice vectors. The second index is the media ID whose maximum value is given by `num_sbe`.

4.26.10 norder_correction

integer, default=0

The order of correction to the electron current density in the Maxwell-SBE calculations.

CHAPTER 5

Release History

5.1 Release Notes

- Release note of SALMON ver. 2:

  https://github.com/SALMON-TDDFT/SALMON2/releases
- Release note of SALMON ver. 0 and 1:

  https://github.com/SALMON-TDDFT/SALMON/releases

5.2 Details of Minor Changes

The following is the history of fixed bugs and changes in models/inputs/outputs after releasing v.2.0.0. (not complete list currently)

5.2.1 Fixed bugs

(Fixed in v.2.2.2)

- Standard output of the DFT part with `convergence/=rho_dne` was incorrect.
- Some external links, such as those to pseudopotential databases and LAPACK libraries, were incorrect.
- `--arch=intel-avx512` was incompatible with the latest versions of Intel compilers.
- Some testsuites (192_bulk_Si_pseudo_pspnc & 724_Si_diamond_bloch_ms) had issues related to compiler dependencies.
- Some `.pspnc` pseudopotentials with certain r-grids caused out-of-bounds errors. Index checking has been updated.

(Fixed in v.2.2.1)

- The OpenACC mode by newer versions of Nvidia HPC SDK was not supported.

- GNU Compiler Collection (GCC) was not supported.
- The sign of the polarizability in isolated systems (yn_periodic=n) was wrong.
- The definition of the excitation energy for the frozen Hamiltonian calculation (yn_fix_func=y) was wrong.
- File output of the atomic force for the multiscale mode (e.g. theory=multi_scale_maxwell_tddft and yn_out_rvf_rt=y) was wrong.
- File output of `sysname_rt.data` for very large systems was wrong.
- An array allocation for the projection_option mode was wrong.
- An array allocation for spin-noncollinear isolated systems (yn_periodic=n and spin=noncollinear) was wrong.

(Fixed in v.2.2.0)

- For the incident pulse for isolated systems (`yn_periodic='n'`), the circular polarization was not defined and the CEP for Acos2 envelope was wrong.
- For spin-unpolarized systems (`spin='unpolarized'`), the initial value of the occupation rate was wrong when the electron number (`nelec`) is odd.
- When `unit_system='A_eV_fs'` is specified, `temperature` was mistakenly defined in the atomic unit.
- For isolated systems (`yn_periodic='n'`), the multipole expansion for the boundary conditions of the Hartree potential (Poisson equation) was wrong.
- For periodic systems (`yn_periodic='y'`), `al_vec[123]` for orthogonal cells yielded unintended error.
- For spin-noncollinear systems (`spin='noncollinear'`), the band-decomposition of the spin magnetization in ***_mag.data files was wrong.

(Fixed in v.2.1.0)

- Non-local term of the transition moment printed by the option of "yn_out_tm=y" has been fixed (just printing issue).
- Parallelization for orbitals for calculation of transition moment by "yn_out_tm=y" has been suported
- Bug of the segmentation fault occurred by "yn_ffte=y" with parallelization for orbitals using isolated system has been fixed.
- Reading of CIF file format for symmetry option has been improved.
- Combination of non-uniform user-defined k-points and symmetry option has been supported.

(Fixed in v.2.0.2)

- Small noise on the total energy in TDDFT calculation (that is seen with weak pulse around e.g. I=1d9 W/cm2) has been removed.
- The printed absolute values of electron density in cube format has been fixed..
- Printing of the external field in TDDFT calculation of the isolated system has a bug in v.2.0.0 and v.2.0.1. It has been fixed in v.2.0.2.
- The file reading option of the external electric field in TDDFT calculation ("file_input1" in &emfield) has been fixed.
- Invalid occupation number printed in SYSNAME_ovlp.data file for projection option with non-uniform k-points has been fixed.

(Fixed in v.2.0.0)

- The imaginary part of wavefunction was not printed in cube format until v.1.2.1

(Fixed in v.?.?.?)

- Abnormal calculation that sometimes happens if zero value is included in the atomic coordinate in the input with "A_eV_fs" has been fiexd.

5.2.2 Changes of models/inputs/outputs

(v.2.2.2)

- Filenames of some output files are changed.
  - `dos.data` –> `<base_directory>/<sysname>_dos.data`
  - `dns.cube` –> `<base_directory>/<sysname>_dns.cube`
  - etc.
- Some input keywords and options are added.
  - `dk_shift`: Shift of the k-vector.
  - `yn_out_rt_energy_components`: Switch for printing out the energy components such as the kinetic energy term (for TDDFT).
  - `magdir_atom`: Initial values for the spin polarization (for DFT).
  - `method_mixing='simple_potential'`: Simple mixing method for the local potential (for DFT).

(v.2.2.1)

- New input keyword for the preconditioning of CG method for accelerating the DFT computation
  - `yn_preconditioning`
- For the spin-noncollinear mode (spin='noncollinear'), some input keywords and output files are changed and added.
  - `sysname_rt_spin.data`: the output file for the spin magnetization and spin current density.
  - Change the input keyword: `out_magnetization_step` –> `out_rt_spin_step`.
  - New input keywords: `yn_out_mag_decomposed_rt`, `yn_out_spin_current_decomposed`, `yn_out_spin_current_micro`.
- New input keyword to read a `.cube` file of the initial electron density for accelerating the DFT computation
  - `method_init_density='read_dns_cube'`

(v.2.2.0)

- New theory options for SBE and Maxwell-SBE are added.
  - `theory = 'sbe'`
  - `theory = 'maxwell_sbe'`
- Input keywords for the Poisson equation solver are added.
  - `method_poisson`
  - `yn_fftw`
- New TDDFT analysis options are added.
  - `yn_fix_func`
  - `projection_option='td'`
  - `threshold_projection`

  - `yn_out_intraband_current`
  - `yn_out_current_decomposed`, `out_current_decomposed_step`
  - `yn_out_micro_je`, `out_micro_je_step`

(v.2.1.0)

- Input variables for Spin-orbit coupling are added
  - "yn_spinorbit"
  - "spin = noncollinear"
  - "out_magnetization_step"
- New options for calculation of dielectric function and conductivity based on transition moments at the end of the GS calculation is added. The related input variables are
  - "yn_out_gs_sgm_eps"
  - "out_gs_sgm_eps_mu_nu"
  - "out_gs_sgm_eps_width"

(v.2.0.2)

- The definition of the total energy of the periodic system printed in TDDFT calculation has changed: The electric field energy is included until v.2.0.1. It has not been included from v.2.0.2.
- The directory names generated by "method_wf_distributor=slice" option have changed from v.2.0.2.

CHAPTER 6

Acknowledgements

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.

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.