Manual for SALMON-v.1.1.0 (simple version)

July 19, 2018

Contents

1 Introduction

1.1 About SALMON

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, Fujitsu FX100 supercomputer system, K-computer, and supercomputer system with Intel Xeon Phi (Knights Landing).

1.2 SALMON features

SALMON describes electron dynamics in both isolated (molecules and nanostructures) and periodic (crystalline solids) systems. 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.

- 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
- 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

1.4 SALMON at Github

SALMON is developed at GitHub.com

1.5 List of developers

(Alphabetic order)

Isabella Floss (TU Wien, Austria)
Yuta Hirokawa (University of Tsukuba, Japan)
Kenji Iida (Institute for Molecular Science, Japan)
Kazuya Ishimura (Institute for Molecular Science, Japan)
Kyung-Min Lee (Max Planck Institute for the Structure and Dynamics of Matter, Germany)
Katsuyuki Nobusada (Institute for Molecular Science, Japan)
Masashi Noda (Institute for Molecular Science, Japan)
Tomohito Otobe (National Institutes for Quantum and Radiological Science and Technology, Japan)
Shunsuke Sato (Max Planck Institute for the Structure and Dynamics of Matter, Germany)
Yasushi Shinohara (University of Tokyo, Japan)
Takashi Takeuchi (Institute for Molecular Science, Japan)
Xiao-Min Tong (University of Tsukuba, Japan)
Mitsuharu Uemoto (University of Tsukuba, Japan)
Kazuhiro Yabana (University of Tsukuba, Japan)
Atsushi Yamada (University of Tsukuba, Japan)
Shunsuke Yamada (University of Tsukuba, Japan)
Maiku Yamaguchi (University of Tokyo, Japan)

1.6 Acknowledgements for SALMON developments

SALMON has been developed by the SALMON developers under supports by Center for Computational Sciences, University of Tsukuba, and Institute for Molecular Science. 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 has been 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.

2 Getting Started (Install & Run)

2.1 Prerequisites

In this guide, it is assumed that readers have a basic knowledge of Unix 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 Fortran/C Compiler
  - Fujitsu Compiler (at FX100 / K-Computer)
- One of the following library packages for linear algebra:
  - BLAS/LAPACK
  - Intel Math Kernel Library (MKL)
  - Fujitsu Scientific Subroutine Library 2 (SSL-II)
- Build tools.
  - CMake

If you use other compilers, you may need to change build scripts (CMake). See . If no numerical library is installed on your computer system, you may need to install BLAS/LAPACK by yourself. See https://salmon-tddft.jp/wiki/Troubleshooting_of_the_Installation_Process.

For the installation of SALMON, we adopt the CMake tools as the first option. If there were any problems to use CMake tools in your environment, you may use the GNU make tools. See https://salmon-tddft.jp/wiki/Troubleshooting_of_the_Installation_Process.

2.2 Download

The newest version of SALMON can be downloaded from https://salmon-tddft.jp/wiki/Download. To extract files from the downloaded file *salmon-.tar.gz*, type the following command in the command-line,

```
$ tar -zxvf ./salmon-.tar.gz
```

After the extraction, the following directories will be created.

```
SALMON
|- src          Source codes
|- example      Samples
|- cmakefiles    CMake related files
|- gnumakefiles    GNU Makefiles for building
```

2.3 Build

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

2.3.1 Checking CMake availability

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

```
$ cmake --version
```

If CMake is not installed in your system, an error message such as `cmake: command not found` will appear. If CMake is installed on your system, the version number will be shown. To build SALMON, CMake of version 3.0.2 or later is required. If you confirm that CMake of version 3.0.2 or later is installed in your system, proceed to . 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

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--.tar.gz`). In standard Unix 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 3.8.2. First get the URL of the download link from your browser, and use `wget` command in your Unix command-line.

```
$ wget https://cmake.org/files/v3.8/cmake-3.8.2-Linux-x86_64.tar.gz
```

Next, unpack the archive by

```
$ tar -zxvf cmake-3.8.2-Linux-x86_64.tar.gz
```

and you will have the binary `make-3.8.2-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-3.8.2-Linux-x86_64/bin:$PATH
```

and then reload the configuration by typing:

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

2.3.3 Build using CMake

Confirming that CMake of version 3.0.2 or later can be usable in your environment, proceed the following steps. We assume that you are in the directory SALMON.

- Create a new temporary directory *build* and move to the directory,

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

- Execute the python script *configure.py* and then make,

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

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

arch	Detail	Compiler	Numerical Library
intel-knl	Intel Knights Landing	Intel Compiler	Intel MKL
intel-knc	Intel Knights Corner	Intel Compiler	Intel MKL
intel-avx	Intel Processer (Ivy-, Sandy-Bridge)	Intel Compiler	Intel MKL
intel-avx2	Intel Processer (Haswell, Broadwell ..)	Intel Compiler	Intel MKL
intel-avx512	Intel Processer (Skylake-SP)	Intel Compiler	Intel MKL
fujitsu-fx100	FX100 Supercomputer	Fujitsu Compiler	SSL-II
fujitsu-k	Fujitsu FX100 / K-computer	Fujitsu Compiler	SSL-II

If the build is successful, you will get a file *salmon.cpu* at the directory *salmon/bin*. If you specify many-core architechtures, *intel-knl* or *intel-knc*, you find a file *salmon.mic* or both files *salmon.cpu* and *salmon.mic*.

2.3.4 Build for single process calculations

In default, the python script assumes parallel execution. If you use a single processor machine, specify `--disable-mpi` in executing the python script:

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

2.3.5 To use Libxc

In SALMON, you may use Libxc functional library, http://www.tddft.org/programs/libxc/installation/. To use the Libxc library, some adittional procedures are necessary. First you need to download the source files in your system as follows:

```
wget http://www.tddft.org/programs/octopus/down.php?file=libxc/4.2.1/libxc-4.2.1.tar.gz
tar -zxvf libxc-4.2.1.tar.gz
```

Then, enter the libxc source directory and make the library as follows:

```
./configure --prefix=INSTALL/PATH/OF/LIBXC
make; make install
```

Finally, enter the SALMON directory and execute `configure.py` script specifying the Libxc directory.

```
configure.py --arch=ARCHITECTURE --prefix=PREFIX --with-libxc=INSTALL/PATH/OF/LIBXC
make; make install
```

2.4 Files necessary to run SALMON

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

2.4.1 Pseudopotentials

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

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

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

2.4.2 input file

Input files are composed of several blocks of namelists,

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

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

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

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

Calculations are usually achieved in two steps; first, the ground state calculation is carried out and then electron dynamics calculations in real time is carried out. A choice of the calculation mode is specified by the variable *calc_mode* in the namelist *&calculation*. For isolated systems, the ground state and the electron dynamics calculations should be carried out as two separate executions. First the ground state calculation is carried out specifying "calc_mode = 'GS' ". Then the real-time electron dynamics calculation is carried out specifying "calc_mode = 'RT' ". For periodic systems, two calculations should be carried out as a single execution specifying "calc_mode = 'GS_RT' ".

In https://salmon-tddft.jp/wiki/Exercises, we prepare six exercises that cover typical calculations feasible by SALMON. We prepare explanations of the input files of the exercises that will help to prepare input files of your own interests.

There are more than 20 groups of namelists. A complete list of namelist variables is given in the file *SALMON/manual/input_variables.md*. Namelist variables that are used in our exercises are explained at https://salmon-tddft.jp/wiki/Input_variables.

2.5 Run SALMON

Before running SALMON, the following preparations are required as described above: The executable file of *salmon.cpu* (and *salmon.mic* if your system is the many-core machine) should be built from the source file of SALMON. An input file *inputfile.inp* and pseudopotential files should also be prepared.

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

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

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

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

where NPROC is the number of MPI processes that you will use. In many-core processor (e.g. intel-knl) environment, the execution command is

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

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

- SALMON runs in both single-process and multi-process environments using MPI.
- executable files are prepared as */salmon/bin/salmon.cpu* and/or */salmon/bin/salmon.mic* in the standard build procedure.
- to start calculations, *inputfile.inp* should be read through *stdin*.

2.6 Appendix

2.6.1 Additional options in configure.py script

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

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

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

Commandline switch	Detail
-a ARCH, --arch=ARCH	Target architecture
--enable-mpi, --disable-mpi	enable/disable MPI parallelization.
--enable-scalapack, --disable-scalapack	enable/disable computations with ScaLAPACK library
--enable-libxc, --with-libxc	See
FC, FFLAGS	User-defined Fortran Compiler, and the compiler options
CC, CFLAGS	User-defined C Compiler, and the compiler options

Build for single process calculations If you use a single processor machine, specify `--disable-mpi` in executing the python script:

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

Build in GCC/GFortran environemnt If you use GCC/GFortran compiler, specify the following flags in executing the python script:

```
$ python ../configure.py FC=gfortran CC=gcc FFLAG=-O3 CFLAG=-O3
```

2.6.2 Build using GNU Makefile

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

```
$ cd SALMON/makefiles
```

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

- fujitsu
- gnu
- gnu-without-mpi
- intel
- intel-avx
- intel-avx2
- intel-knc
- intel-knl
- 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/`.

3 Inputs

We here summarize namelists that appear in this Tutorial. A thorough list of the namelist variables may be found in the downloaded file in 'SALMON/manual/input_variables.md'.

3.1 &units

Mandatory: none

```
&units
  unit_system='A_eV_fs'
/
```

This namelist specifies the unit system to be used in the input file. Options are 'A_eV_fs' for Angstrom, eV, and fs, and 'a.u.' or 'au' for atomic units. If you do not specify it, atomic unit will be used as default.

For isolated systems (specified by `iperiodic = 0` in `&system`), the unit of 1/eV is used for the output files of DOS and PDOS if `unit_system = 'A_eV_fs'` is specified, while atomic unit is used if not. For other output files, the Angstrom/eV/fs units are used irrespective of the namelist value.

For periodic systems (specified by `iperiodic =3` in `&system`), the unit system specified by this namelist variable is used for most output files. See the first few lines of output files to confirm the unit system adopted in the file.

3.2 &calculation

Mandatory: calc_mode

```
&calculation
  calc_mode = 'GS'
/
```

The value of the `calc_mode` should be one of `'GS'`, `'RT'`, and `'GS-RT'`. For isolated systems (specified by `iperiodic = 3` in `&system`), the ground state (`'GS'`) and the real time (`'RT'`) calculations should be done separately and sequentially. For periodic systems (specified by `iperiodic = 3` in &system), both ground state and real time calculations should be carried out as a single task (calc_mode = 'GS_RT').

For Maxwell + TDDFT multi-scale calculation, add the following namelist.

```
use_ms_maxwell = 'y'
```

3.3 &control

Mandatory: none

```
&control
  sysname = 'C2H2'
/
```

'C2H2' defined by `sysname = 'C2H2'` will be used in the filenames of output files. If you do not specify it, the file name will start with 'default'.

3.4 &functional

```
&functional
  xc ='PZ'
/
```

`xc ='PZ'` indicates that (adiabatic) local density approximation is adopted (Perdew-Zunger: Phys. Rev. B23, 5048 (1981)). This is the default choice.

For isolated systems (specified by `iperiodic = 0` in `&system`), only the default choice of 'PZ' is available at present.

For periodic systems (specified by `iperiodic = 3` in `&system`), the following functionals may be available in addition to 'PZ':

```
xc = 'PZM'
```

Perdew-Zunger LDA with modification to improve sooth connection between high density form and low density one. :J. P. Perdew and Alex Zunger, Phys. Rev. B 23, 5048 (1981).

`xc = 'TBmBJ' cval = 1.0`

Tran-Blaha meta-GGA exchange with Perdew-Wang correlation. :Fabien Tran and Peter Blaha, Phys. Rev. Lett. 102, 226401 (2009). John P. Perdew and Yue Wang, Phys. Rev. B 45, 13244 (1992). This potential is known to provide a reasonable description for the bandage of various insulators. For this choice, the additional mixing parameter 'cval' may be specified. If cval is set to a minus value, the mixing-parameter will be computed following the formula in the original paper [Phys. Rev. Lett. 102, 226401 (2009)]. The default value for this parameter is 1.0.

Since version 1.1.0, exchange-correlation functionals in Libxc library (http://www.tddft.org/programs/libxc/) have been usable in SALMON. At present, usable functionals are limited to LDA and GGA. For periodic systems, meta-GGA functionals are usable as well. To specify the exchange-correlation potentials of Libxc library, there are two ways. If the exchange and correlation potentials are given separately, you need to specify both `alibx` and `alibc` separately. If the exchange and correlation potentials are given as a combined set, you need to specify `alibxc`. We show below an example:

```
&functional
alibx = 'LDA_X'
alibc = 'LDA_C_PZ'
/
```

Available sets of the functionals are listed at the website http://www.tddft.org/programs/libxc/functionals/ .

Note that, the hybrid functionals (hybrid gga/mgga) are not supported in the current (version 1.1.0) of SALMON.

3.5 &system

Mandatory: iperiodic, al, nstate, nelem, natom

For an isolated molecule (Tutorial-1, 2, 3):

```
&system
  iperiodic = 0
  al = 16d0, 16d0, 16d0
  nstate = 5
  nelem = 2
  natom = 4
  nelec = 10
/
```

`iperiodic = 0` indicates that the isolated boundary condition will be used in the calculation. `al = 16d0, 16d0, 16d0` specifies the lengths of three sides of the rectangular parallelepiped where the grid points are prepared. `nstate = 8` indicates the number of Kohn-Sham orbitals to be solved. `nelec = 10` indicate the number of valence electrons in the system. Since the present code assumes that the system is spin saturated, `nstate` should be equal to or larger than `nelec/2`. `nelem = 2` and `natom = 4` indicate the number of elements and the number of atoms in the system, respectively.

For a periodic system (Tutorial-4, 5):

```
&system
  iperiodic = 3
  al = 10.26d0,10.26d0,10.26d0
  nstate = 32
  nelec = 32
  nelem = 1
  natom = 8
/
```

`iperiodic = 3` indicates that three dimensional periodic boundary condition (bulk crystal) is assumed. `al = 10.26d0, 10.26d0, 10.26d0` specifies the lattice constans of the unit cell. `nstate = 32`

indicates the number of Kohn-Sham orbitals to be solved. `nelec = 32` indicate the number of valence electrons in the system. `nelem = 1` and `natom = 8` indicate the number of elements and the number of atoms in the system, respectively.

For Maxwell - TDDFT multi scale calculation (Tutorial-6):

```
&system
  iperiodic = 3
  al = 10.26d0,10.26d0,10.26d0
  isym = 8
  crystal_structure = 'diamond'
  nstate = 32
  nelec = 32
  nelem = 1
  natom = 8
/
```

The difference from the above case is the variables, `isym = 8` and `crystal_structure = 'diamond'`, which indicates that the spatial symmetry of the unit cell is used in the calculation. Although the use of the symmetry substantially reduces the computational cost, it should be used very carefully. At present, the spatial symmetry has been implemented only for the case of the diamond structure.

3.6 &pseudo

Mandatory: pseudo_file, izatom

For C2H2 molecule:

```
&pseudo
  izatom(1)=6
  izatom(2)=1
  pseudo_file(1)='C_rps.dat'
  pseudo_file(2)='H_rps.dat'
  lmax_ps(1)=1
  lmax_ps(2)=0
  lloc_ps(1)=1
  lloc_ps(2)=0
/
```

Parameters related to atomic species and pseudopotentials. `izatom(1) = 6` specifies the atomic number of the element #1. `pseudo_file(1) = 'C_rps.dat'` indicates the filename of the pseudopotential of element #1. `lmax_ps(1) = 1` and `lloc_ps(1) = 1` specify the maximum angular momentum of the pseudopotential projector and the angular momentum of the pseudopotential that will be treated as local, respectively.

For crystalline Si:

```
&pseudo
  izatom(1)=14
  pseudo_file(1) = './Si_rps.dat'
  lloc_ps(1)=2
/
```

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

3.7 &atomic_coor

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

For C2H2 molecule:

```
&atomic_coor
  'C' 0.000000 0.000000 0.599672 1
  'H' 0.000000 0.000000 1.662257 2
  'C' 0.000000 0.000000 -0.599672 1
  'H' 0.000000 0.000000 -1.662257 2
/
```

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

3.8 &atomic_red_coor

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

For a crystalline silicon:

```
&atomic_red_coor
  'Si' .0 .0 .0 1
  'Si' .25 .25 .25 1
  'Si' .5 .0 .5 1
  'Si' .0 .5 .5 1
  'Si' .5 .5 .0 1
  'Si' .75 .25 .75 1
  'Si' .25 .75 .75 1
  'Si' .75 .75 .25 1
/
```

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

3.9 &rgrid

Mandatory: dl or num_rgrid

This namelist provides grid spacing of Cartesian coordinate system. `dl(3)` specify the grid spacing in three Cartesian coordinates. This is adopted for C2H2 calculation (Tutorial-1).

```
&rgrid
dl = 0.25d0, 0.25d0, 0.25d0
/
```

`num_rgrid(3)` specify the number of grid points in each Cartesian direction. This is adopted for crystalline Is calculation (Tutorial-4, 5, 6).

```
&rgrid
  num_rgrid = 12,12,12
/
```

3.10 &kgrid

Mandatory: none

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

```
&kgrid
num_kgrid = 4,4,4
/
```

3.11 &scf

Mandatory: nscf

This namelists specify parameters related to the self-consistent field calculation.

```
&scf
  ncg = 4
  nscf = 1000
  convergence = 'norm_rho_dng'
  threshold_norm_rho = 1.d-15
/
```

`ncg = 4` is the number of conjugate-gradient iterations in solving the Kohn-Sham equation. Usually this value should be 4 or 5. `nscf = 1000` is the number of scf iterations. For isolated systems specified by `&system/iperiodic = 0`, the scf loop in the ground state calculation ends before the number of the scf iterations reaches `nscf`, if a convergence criterion is satisfied. There are several options to examine the convergence. If the value of `norm_rho_dng` is specified, the convergence is examined by the squared difference of the electron density,

3.12 &hartree

Mandatory: none

```
&hartree
  meo = 3
  num_pole_xyz = 2,2,2
/
```

`meo` specifies the order of multipole expansion of electron density that is used to prepare boundary condition for the Hartree potential.

- `meo=1`: Monopole expansion (spherical boundary condition).
- `meo=2`: Multipole expansions around each atom.
- `meo=3`: Multipole expansion around the center of mass of electrons in cubits that are defined by `num_pole_xyz`.

`num_pole_xyz(3)` defines the division of space when `meo = 3` is specified.

A default for `meo` is `3`, and defaults for `num_pole_xyz` are `(0,0,0)`. When default is set for `num_pole_xyz`, the division of space is carried out using a prescribed method.

3.13 &tgrid

Mandatory: dt, Nt

```
&tgrid
  dt=1.25d-3
  nt=5000
/
```

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

3.14 &propagation

This namelist specifies the numerical method for time evolution calculations of electron orbitals.

```
&propagation
  propagator='etrs'
/
```

`propagator = 'etrs'` indicates the use of enforced time-reversal symmetry propagator. M.A.L. Marques, A. Castro, G.F. Bertsch, and A. Rubio, Comput. Phys. Commun., 151 60 (2003).

```
&propagation
  propagator='middlepoint'
/
```

`propagation='middlepoint'` indicates that Hamiltonian at midpoint of two-times is used.
The default is *middlepoint*.

3.15 &emfield

This namelist specifies the pulse shape of an electric filed applied to the system in time evolution calculations. We explain below separating two cases, #Linear response calculations and #Pulsed electric field calculations.

3.15.1 Linear response calculations

A weak impulsive field is applied at *t=0*. For this case, `ae_shape1 = 'impulse'` should be described.
Mandatory: ae_shape1

```
&emfield
  ae_shape1 = 'impulse'
  epdir_re1 = 0.d0,0.d0,1.d0
/
```

`epdir_rel(3)` specify a unit vector that indicates the direction of the impulse.
For a periodic system specified by `iperiodic = 3`, one may add `trans_longi`. It has the value, `'tr'`(transverse) or `'lo'`(longitudinal), that specifies the treatment of the polarization in the time evolution calculation. The default is `'tr'`.

```
&emfield
  trans_longi = 'tr'
  ae_shape1 = 'impulse'
  epdir_re1 = 0.,0.,1.
/
```

The magnitude of the impulse of the pulse may be explicitly specified by, for example, `e_impulse = 1d-2`. The default is '1d-2' in atomic unit.

3.15.2 Pulsed electric field calculations

A Pulsed electric field of finite time duration is applied. For this case, `as_shape1` should be specified. It indicates the shape of the envelope of the pulse. The options include 'Acos2' and 'Ecos2' (See below for other options).
Mandatory: ae_shape1, epdir_re1, {rlaser_int1 or amplitude1}, omega1, pulse_tw1, phi_cep1

```
&emfield
  ae_shape1 = 'Ecos2'
  epdir_re1 = 0.d0,0.d0,1.d0
  rlaser_int_wcm2_1 = 1.d8
  omega1=9.28d0
  pulse_tw1=6.d0
  phi_cep1=0.75d0
/
```

`ae_shape1 = 'Ecos2'` specifies the envelope of the pulsed electric field, 'Ecos2' for the cos^2 envelope for the electric field. If 'Acos2' is specified, this gives cos^2 envelope for the vector potential. Note that 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish. There is no such restriction for the 'Acos2' pulse shape.

`epdir_re1 = 0.d0,0.d0,1.d0` specifies the real part of the unit polarization vector of the pulsed electric field. If only the real part is specified, it describes a linearly polarized pulse. Using both real ('epdir_re1') and imaginary ('epdir_im1') parts of the polarization vector, circularly (and general ellipsoidary) polarized pulses may be described.

`laser_int_wcm2_1 = 1.d8` specifies the maximum intensity of the applied electric field in unit of W/cm^2. It is also possible to specify the maximum intensity of the pulse by `amplitude1`.

`omega1=9.26d0` specifies the average photon energy (frequency multiplied with hbar).

`pulse_tw1=6.d0` specifies the pulse duration. Note that it is not the FWHM but a full duration of the cos^2 envelope.

`phi_cep1=0.75d0` specifies the carrier envelope phase of the pulse. As noted above, 'phi_cep1' must be 0.75 (or 0.25) if one employs 'Ecos2' pulse shape, since otherwise the time integral of the electric field does not vanish. There is no such restriction for the 'Acos2' pulse shape.

It is possible to use two pulses simultaneously to simulate pump-probe experiments, adding information for two pulses. To specify the second pulse, change from 1 to 2 in the namelist variables, like `ae_shape2`. The time delay between two pulses is specified by the variable 't1_t2'.

For a periodic system specified by `iperiodic = 3`, one may add `trans_longi`. It has the value, `'tr'`(transverse) or `'lo'`(longitudinal), that specifies the treatment of the polarization in the time evolution calculation. The default is `'tr'`. For a periodic system, it is also specify 'Acos3', 'Acos4', 'Acos6', 'Acos8' for `ae_shape1`.

3.16 &analysis

Mandatory: none

The following namelists specify whether the output files are created or not after the calculation. In the ground state calculation of isolated systems (Tutorial-1):

```
&analysis
  out_psi = 'y'
  out_dns = 'y'
  out_dos = 'y'
  out_pdos = 'y'
  out_elf = 'y'
/
```

In the time evolution calculation of isolated systems (Tutorial-3):

```
&analysis
  out_dns_rt = 'y'
  out_elf_rt = 'y'
  out_estatic_rt = 'y'
/
```

In the following namelists, variables related to time-frequency Fourier analysis are specified.

```
&analysis
  nenergy=1000
  de=0.001
/
```

`nenergy=1000` specifies the number of energy steps, and `de=0.001` specifies the energy spacing in the time-frequency Fourier transformation.

3.17 &multiscale

This namelist specifies information necessary for Maxwell - TDDFT multiscale calculations.

```
&multiscale
  fdtddim = '1D'
  twod_shape = 'periodic'
  nx_m = 4
  ny_m = 1
  hX_m = 250d0
  nxvacl_m = -2000
  nxvacr_m = 256
/
```

`fdtddim` specifies the spatial dimension of the macro system. `fdtddim='1D'` indicates that one-dimensional equation is solved for the macroscopic vector potential.

`nx_m = 4` specifies the number of the macroscopic grid points in for x-direction in the spatial region where the material exists.

`hx_m = 250d0` specifies the grid spacing of the macroscopic grid in x-direction.

`nxvacl_m = -2000` and `nxvacr_m = 256` indicate the number of grid points in the vacuum region, `nxvacl_m` for the left and `nxvacr_m` for the right from the surface of the material.

3.18 &parallel

When you execute a job with MPI parallelization, you are not required to specify any parameters that describe the assignment of the parallelization; the assignment is carried out automatically. You may also specify the parameters explicitly as below.

Mandatory: none

```
&parallel
  nproc_ob = 1
  nproc_domain = 1,1,1
  nproc_domain_s = 1,1,1
/
```

- `nproc_ob` specifies the number of MPI parallelization to divide the electron orbitals. The default value is 0 (automatic parallelization).
- `nproc_domain(3)`specifies the number of MPI parallelization to divide the spatial grids of the electron orbitals in three Cartesian directions. The default values are (0/0/0) (automatic parallelization).
- `nproc_domain_s(3)'` specifies the number of MPI parallelization to divide the spatial grids related to the electron density in three Cartesian directions. The default values are (0/0/0) (automatic parallelization).

The following conditions must be satisfied.

- The total number of processors must be equal to both `nproc_ob * nproc_domain(1) * nproc_domain(2) * nproc_domain(3)` and also `nproc_domain_s(1) * nproc_domain_s(2) * nproc_domain_s(3)`.
- `nproc_domain_s(1)` is a multiple of `nproc_domain(1)`, and the same relations to the second and third components.

4 Exercises

4.1 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, https://salmon-tddft.jp/wiki/download and https://salmon-tddft.jp/wiki/Install_and_Run.

As described in https://salmon-tddft.jp/wiki/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 namelist variables 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 6 exercises.

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

Next 2 exercises (Exercise-4 ~ 5) are for a crystalline solid, silicon. If you are interested in learning electron dynamics calculations in extended periodic systems, please look into these exercises. Since ground state calculations of small unit-cell systems are not computationally expensive and a time evolution calculation is usually much more time-consuming than the ground state calculation, we recommend to run the ground and the time evolution calculations as a single job. The following two exercises are organized in that way. Exercise-4 illustrates the calculation of linear response properties of crystalline silicon to obtain the dielectric function. Exercise-5 illustrates the calculation of electron dynamics in the crystalline silicon induced by a pulsed electric field.

The final exercise (Exercise-6) is for an irradiation and a propagation of a pulsed light in a bulk silicon, coupling Maxwell equations for the electromagnetic fields of the pulsed light and the electron dynamics in the unit cells. This calculation is quite time-consuming and is recommended to execute using massively parallel supercomputers. Exercise-6 illustrates the calculation of a pulsed, linearly polarized light irradiating normally on a surface of a bulk silicon.

4.2 C2H2 (isolated molecules)

4.2.1 Exercise-1: Ground state of C2H2 molecule

In this exercise, we learn the calculation of the ground state solution 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. It should be noted that at present it is not possible to carry out the geometry optimization in SALMON. Therefore, atomic positions of the molecule are specified in the input file and are fixed during the calculations.

Input files To run the code, following files are used:

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

You may download the above 3 files (zipped file) from:
https://salmon-tddft.jp/wiki/File:C2H2_gs_input.zip

In the input file *C2H2_gs.inp*, namelists variables are specified. Most of them are mandatory to execute the ground state calculation. We present their explanations below:
https://salmon-tddft.jp/wiki/Explanations_of_input_files_(ground_state_of_C2H2_molecule)

This will help you to prepare an input file for other systems that you want to calculate. A complete list of the namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
C2H2_info.data	information on ground state solution
dns.cube	a cube file for electron density
elf.cube	electron localization function (ELF)
psi1.cube, *psi2.cube*, ...	electron orbitals
dos.data	density of states
pdos1.data, *pdos2.data*, ...	projected density of states
C2H2_gs.bin	binary output file to be used in the real-time calculation

You may download the above files (zipped file, except for the binary file *C2H2_gs.bin*) from:
https://salmon-tddft.jp/wiki/File:C2H2_gs_output.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 described in:
https://salmon-tddft.jp/wiki/Explanations_of_output_files_(ground_state_of_C2H2_molecule)

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

image	files used to create the image
highest occupied molecular orbital (HOMO)	*psi1.cube*, *psi2.cube*, ...
electron density	*dns.cube*
electron localization function	*elf.cube*

4.2.2 Exercise-2: Polarizability and photoabsorption of C2H2 molecule

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

Input files To run the code, the input file *C2H2_rt_response.inp* that contains namelist variables and their values for the linear response calculation is required. The binary file *C2H2_gs.bin* that is created in the ground state calculation and pseudopotential files are also required. The pseudopotential files should be the same as those used in the ground state calculation.

file name	description
C2H2_rt_response.inp	input file that contains namelist variables and their values
C_rps.dat	pseodupotential file for carbon
H_rps.dat	pseudopotential file for hydrogen
C2H2_gs.bin	binary file created in the ground state calculation

You may download the *C2H2_rt_response.inp* file (zipped file) from:
https://salmon-tddft.jp/wiki/File:C2H2_rt_response_input.zip
In the input file *C2H2_rt_response.inp*, namelists variables are specified. Most of them are mandatory to execute the linear response calculation. We present their explanations below:
https://salmon-tddft.jp/wiki/Explanations_of_input_files_(polarizability_and_photoabsorption_of_C2H2

This will help you to prepare the input file for other systems that you want to calculate. A complete list of the namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
C2H2_lr.data	polarizability and oscillator strength distribution as functions of energy
C2H2_p.data	components of dipole moment as functions of time

You may download the above files (zipped file) from:
https://salmon-tddft.jp/wiki/File:C2H2_rt_response_output.zip
Explanations of the output files are given in:
https://salmon-tddft.jp/wiki/Explanations_of_output_files_(polarizability_and_photoabsorption_of_C2H

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

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

Input files To run the code, following files are used. The *C2H2_gs.bin* file is created in the ground state calculation. Pseudopotential files are already used in the ground state calculation. Therefore, *C2H2_rt_pulse.inp* that specifies namelist variables and their values for the pulsed electric field calculation is the only file that the users need to prepare.

file name	description
C2H2_rt_pulse.inp	input file that contain namelist variables and their values.
C_rps.dat	pseodupotential file for Carbon
H_rps.dat	pseudopotential file for Hydrogen
C2H2_gs.bin	binary file created in the ground state calculation

You may download the *C2H2_rt_pulse.inp* file (zipped file) from:
https://salmon-tddft.jp/wiki/File:C2H2_rt_pulse_input.zip
In the input file *C2H2_rt_pulse.inp*, namelists variables are specified. Most of them are mandatory to execute the calculation of electron dynamics induced by a pulsed electric field. We present explanations of the namelist variables that appear in the input file in:
https://salmon-tddft.jp/wiki/Explanations_of_input_files_(C2H2_molecule_under_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 namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
C2H2_p.data	components of the electric dipole moment as functions of time
C2H2_ps.data	power spectrum that is obtained by a time-frequency Fourier transformation of the electric dipole mome

You may download the above files (zipped file) from:
https://salmon-tddft.jp/wiki/File:C2H2_rt_pulse_output.zip
Explanations of the files are described in:
https://salmon-tddft.jp/wiki/Explanations_of_output_files_(C2H2_molecule_under_a_pulsed_electric_fie

4.3 Crystalline silicon (periodic solids)

4.3.1 Exercise-4: Dielectric function of crystalline silicon

In this exercise, we learn the linear response calculation of the crystalline silicon of a diamond structure. Calculation is done in a cubic unit cell that contains eight silicon atoms. Since the ground state calculation costs much less computational time than the time evolution calculation, both calculations are successively executed. After finishing the ground state calculation, an impulsive perturbation is applied to all electrons in the unit cell along z direction. Since the dielectric function is isotropic in the diamond structure, calculated dielectric function should not depend on the direction of the perturbation. During the time evolution, electric current averaged over the unit cell volume is calculated. A time-frequency Fourier transformation of the electric current gives us a frequency-dependent conductivity. The dielectric function may be obtained from the conductivity using a standard relation.

Input files To run the code, following files are used:

file name	description
Si_gs_rt_response.inp	input file that contain namelist variables and their values.
Si_rps.dat	pseodupotential file of silicon

You may download the above 2 files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_response_input.zip
In the input file *Si_gs_rt_response.inp*, namelists variables are specified. Most of them are mandatory to execute the calculation. We present explanations of the namelist variables that appear in the input file in:
https://salmon-tddft.jp/wiki/Explanations_of_input_files_(dielectric_function_of_crystalline_silicon
This will help you to prepare the input file for other systems that you want to calculate. A complete list of the namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
Si_gs_info.data	information of ground state calculation
Si_eigen.data	energy eigenvalues of orbitals
Si_k.data	information on k-points
Si_rt.data	electric field, vector potential, and current as functions of time
Si_force.data	force acting on atoms
Si_lr.data	Fourier spectra of the dielectric functions
Si_gs_rt_response.out	standard output file

You may download the above files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_response_output.zip
Explanations of the output files are described in:
https://salmon-tddft.jp/wiki/Explanation_of_output_fiels_(dielectric_function_of_crystalline_silicon

4.3.2 Exercise-5: Electron dynamics in crystalline silicon under a pulsed electric field

In this exercise, we learn the calculation of electron dynamics in a unit cell of crystalline silicon of a diamond structure. Calculation is done in a cubic unit cell that contains eight silicon atoms. Since the ground state calculation costs much less computational time than the time evolution calculation, both calculations are successively executed. After finishing the ground state 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, and carrier envelope phase are specified in the input file.

Input files To run the code, following files are used:

file name	description
Si_gs_rt_pulse.inp	input file that contain namelist variables and their values.
Si_rps.dat	pseodupotential file for Carbon

You may download the above 2 files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_pulse_input.zip
In the input file *Si_gs_rt_pulse.inp*, namelists variables are specified. Most of them are mandatory to execute the calculation. We present explanations of the namelist variables that appear in the input file in:
https://salmon-tddft.jp/wiki/Explanation_of_input_files_(crystalline_silicon_under_a_pulsed_electric
This will help you to prepare the input file for other systems that you want to calculate. A complete list of the namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
Si_gs_info.data	information of ground state calculation
Si_eigen.data	energy eigenvalues of orbitals
Si_k.data	information on k-points
Si_rt.data	electric field, vector potential, and current as functions of time
Si_force.data	force acting on atoms
Si_lr.data	Fourier transformations of various quantities
Si_gs_rt_pulse.out	standard output file

You may download the above files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_pulse_output.zip
Explanations of the output files are described in:
https://salmon-tddft.jp/wiki/Explanation_of_output_files_(crystalline_silicon_under_a_pulsed_electri

4.4 Maxwell + TDDFT multiscale simulation

4.4.1 Exercise-6: Pulsed-light propagation through a silicon thin film

In this exercise, we learn the calculation of the propagation of a pulsed light through a thin film of crystalline silicon. We consider a silicon thin film of 53 nm thickness, and an irradiation of a few-cycle, linearly polarized pulsed light normally on the thin film. First, to set up initial orbitals, the ground state calculation is carried out. The pulsed light locates in the vacuum region in front of the thin film. The parameters that characterize the pulsed light such as magnitude and frequency are specified in the input file. The calculation ends when the reflected and transmitted pulses reach the vacuum region.

Input files To run the code, following files are used:

file name	description

Si_gs_rt_multiscale.inp	input file that contain namelist variables and their values.
Si_rps.dat	pseodupotential file for silicon

You may download the above two files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_multiscale_input.zip
In the input file *Si_gs_rt_multiscale.inp*, namelists variables are specified. Most of them are mandatory to execute the calculation. We present explanations of the namelist variables that appear in the input file in:
https://salmon-tddft.jp/wiki/Explanation_of_input_files_(pulsed-light_propagation_through_a_silicon_
This will help you to prepare the input file for other systems that you want to calculate. A complete list of the namelist variables that can be used in the input file can be found in the downloaded file *SALMON/manual/input_variables.md*.

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

file name	description
Si_gs_info.data	results of the ground state as well as input parameters
Si_eigen.data	orbital energies in the ground state calculation
Si_k.data	information on k-points
RT_Ac/Si_Ac_xxxxxx.data	various quantities at a time as functions of macroscopic position
RT_Ac/Si_Ac_vac.data	vector potential at vacuum position adjacent to the medium
Mxxxxx/Si_Ac_M.data	various quantities at a macroscopic point as functions of time
Si_gs_rt_multiscale.out	standard output file

You may download the above files (zipped file) from:
https://salmon-tddft.jp/wiki/File:Si_gs_rt_multiscale_output.zip
Explanations of the output files are described in:
https://salmon-tddft.jp/wiki/Explanation_of_output_files_(pulsed-light_propagation_through_a_silicon

5 References

5.1 Suggested citations

If you publish a paper in which SALMON makes an important contribution, we suggest you to cite the following articles.

- If you use SALMON for electron dynamics calculations of a large-size system, the following paper that discusses massively parallel implementation utilizing spatial divisions will be appropriate:

M. Noda et.al, Massively-parallel electron dynamics calculations in real-time and real-space: toward applications to nanostructures of more than ten-nanometers in size, J. Compute. Phys., 2014, 265, 145-155.

- if you use SALMON to calculate electron dynamics in a unit cell of crystalline solid, the following paper discussing formalism and numerical implementation will be appropriate:

G.F. Bertsch et.al, Real-space, real-time method for the dielectric function, Phys. Rev. B62, 7998 (2000).

- The following paper is one of the first implementations of the real-time time-dependent density functional calculation, in particular, instantaneous kick for the linear response calculations:

K. Yabana et.al, Phys. Rev. B54, 4484 (1996).

- If you use multiscale calculation coupling Maxwell equations for the electromagnetic fields of light and electron dynamics, the following paper discussing the formalism and the numerical implementation will be appropriate:

K. Yabana et.al, Time-dependent density functional theory for strong electromagnetic fields in crystalline solids, Phys. Rev. B85, 045134 (2012).

- The following paper describes parallelization method for the coupled Maxwell - TDDFT calculations:

S.A. Sato et.al, Maxwell + TDDFT multi-scale simulation for laser-matter interaction, J. Adv. Simulat. Sci. Eng. 1, 98 (2014).

- The following paper describes computational aspects of electron dynamics calculations for periodic systems in many-core processors:

Y. Hirokawa et.al, Electron dynamics simulation with time-dependent density functional theory on large scale symmetric mode Xeon Phi Cluster, IEEE IPDPS Workshop PDSEC'16 (2016).

6 References