TeNeS Documentation

Release 0.1

Institute for Solid State Physics, University of Tokyo

Dec 04, 2019

CONTENTS

CHAPTER

ONE

WHAT IS TENES ?

1.1 Overview

TeNeS is an open-source program package for calculation of many-body quantum states base on the tensor network method. In ver. 0.1, this package calculates ground-state wavefunctions of quantum spin models, and evaluates physical quantities such as magnetization and correlation functions. For predefined models and lattices, there is a tool that makes it easy for users to generate input files. In ver. 1.0, a tool that is useful to define the own models and lattices from a simple input file will be developed. Furthermore, the Boson system will be able to be treated. This improvement will enable us to deal with various two-dimensional quantum spin and boson systems.

1.2 Developers

TeNeS is developed by the following members.

- ver 0.1
  - Tsuyoshi Okubo (Graduate School of Science, Univ. of Tokyo)
  - Satoshi Morita (Institute for Solid State Physics, Univ. of Tokyo)
  - Yuichi Motoyama (Institute for Solid State Physics, Univ. of Tokyo)
  - Kazuyoshi Yoshimi (Institute for Solid State Physics, Univ. of Tokyo)
  - Takeo Kato (Institute for Solid State Physics, Univ. of Tokyo)
  - Naoki Kawashim (Institute for Solid State Physics, Univ. of Tokyo)

1.3 Version information

- ver. 0.1: 2019/12/04.

1.4 License

1.5 Copyright

This software was developed with the support of "*Project for advancement of software usability in materials science*" of The Institute for Solid State Physics, The University of Tokyo.

CHAPTER

TWO

INSTALL

2.1 Download

You can download the source code for‘‘TeNeS‘‘from the GitHub page . If you have git installed on your machine, type the following command to start download:

```
$ git clone https://github.com/issp-center-dev/TeNeS
```

2.2 Prerequisites

The following tools are required for building TeNeS.

1. C++11 compiler
2. CMake (>=2.8.14)
3. MPI and ScaLAPACK

`TeNeS` depends on the following libraries, but these are downloaded automatically through the build process.

1. mptensor
2. cpptoml
3. sanitizers-cmake

ScaLAPACK must be installed by yourself. If you use homebrew in Mac, type the following command:

```
brew install scalapack
```

For others, see the web page of ScaLAPACK.

For `tenes_simple` which generates the input file for `tenes`, the following libraries are needed.

1. Python (>= ver.3 is recommended)
2. numpy
3. toml

2.3 Install

1. Build `TeNeS` by typing the following commands:

```
$ mkdir build
$ cd build
$ cmake  -DCMAKE_INSTALL_PREFIX=<path to install to> ..
$ make
```

The executable file `tests` will be generated in `build/src` directory. The default value of the `<path to install to>` is `/usr/local`.

2. Install `TeNeS` by typing the following commands:

```
$ make install
```

In this case, `tenes` is installed into the `<path to install to>/bin`.

Specify compiler

`CMake` detects your compiler automatically but sometimes this is not what you want. In this case, you can specify the compiler by the following way,

```
$ cmake -DCMAKE_CXX_COMPILER=<path to your compiler> ../
```

Use the pre-built mptensor

`TeNeS` is based on the parallerized tensor library `mptensor`. The build system of `TeNeS` installs this automatically, but if you want to use the specific version of the mptensor, please add the following option in cmake.

```
$ cmake -DMPTENSOR_ROOT=<path to mptensor> ../
```

CHAPTER

THREE

USAGE

TeNeS needs to create several input files to define models, order of operations, etc. You can create and run the input file directly, but the following script is provided for ease of use:

- `tenes_simple.py`: A tool that generates input files for `tenes` by using a simple input file which specifies lattice model predefined.
- `tenes_standard.py` (from ver. 1.0): A tool that generates input files for `tenes` by using input files for generating the lattice model.

The following sections describe how to use each script, and finally how to use `tenes`. If you want to work with other models or grids, you can do so by creating the input file for `tenes` directly. See *File format* for details on the input file of `tenes`.

3.1 Usage of `tenes_simple`

`tenes_simple` is a tool that creates an input file of `tenes` for predefined models, lattices.

```
$ ./tenes_simple --help
usage: tenes_simple [-h] [-o OUTPUT] input

Simple input generator for TeNeS

positional arguments:
  input                 Input TOML file

optional arguments:
  -h, --help            show this help message and exit
  -o OUTPUT, --output OUTPUT
                        Output TOML file
```

- Take the input file name as an argument.
- The command line options are:
  - **help**
    - show help messages.
  - **output**
    - output file name.
    - default name is `input.toml`.
    - Cannot have the same file name as the input file name.

3.2 Usage of `tenes_simple.py`

`tenes_simple.py` will be released at ver.1.0.

3.3 Usage of `tenes`

If you prepare input files of `tenes` by yourself, You can directry run `tenes` as follows:

```
$ tenes --help
TeNeS: TEnsor NEtwork Solver for 2D quantum lattice system

  Usage:
    tenes [--quiet] <input_toml>
    tenes --help
    tenes --version

  Options:
    -h --help       Show this help message.
    -v --version    Show the version.
    -q --quiet      Do not print any messages.
```

- Take the input file name as an argument.
- The command line options are:
  - `help` - Show help messages.
  - `version` - Show the version number.
  - `quiet` - Do not print any messages to the standard output.

```
tnsolve input.toml
```

CHAPTER

FOUR

TUTORIAL

4.1 Ising model with transverse magnetic field

This section presents an example of calculation when the transverse magnetic field is applied to the Ising model. By changing the variable `G` in the input file, the magnitude of the transverse magnetic field will be modified. For example, when the transverse magnetic field is 0, the input file becomes

```
[parameter]
[parameter.tensor]
D  = 2     # tensor_dim
CHI  = 10  # env_dim

[parameter.simple_update]
num_step = 1000
tau = 0.01

[parameter.ctm]
iteration_max = 10

[lattice]
type = "square lattice"
L_sub = [2,2]

[model]
type = "spin"
Jz = -1.0
Jx = 0.0
Jy = 0.0
G = 0.0
```

In this case, since `Jz = -1.0` , the ferro magnetic state becomes the ground state at `G=0`. When the input file name is `simple.toml` , type the following commands to execute `tenes` :

```
$ tenes_simple simple.toml
$ tenes input.toml
```

Then, the following logs are output to the standard output.

```
Start simple update
Start calculating observables
Start updating environment
Start calculating local operators
Save site observables to output/site_obs.dat
```

(continues on next page)

(continued from previous page)

```
Start calculating energy
Save energy to output/energy.dat
Start calculating NN correlation
Save NN correlation to output/neighbor_obs.dat
Save elapsed times to output/time.dat

Energy = -0.5
Local operator 0 = 0.5
Local operator 1 = 1.90794709356e-11

time simple update = 3.21127
time full update   = 0
time environmnent  = 0.875561
time observable    = 0.132412
```

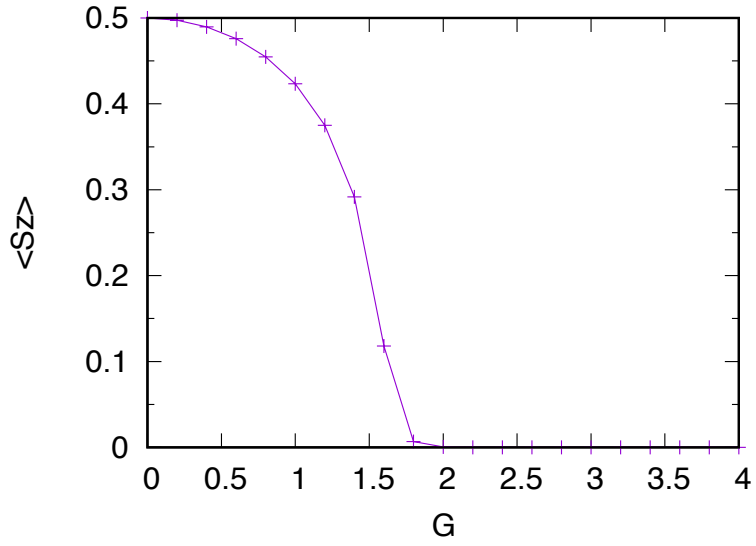
The execution status of each process is displayed first. After finishing the calculation, `Energy` and the local operators `Local operator 0` (<Sz>), `Local operator 1` (<Sx>) are outputted, respectively. Finally, `time` is printed to show the calculation time (Time unit is seconds). In the `output` folder, `energy.dat, parameters.dat, time.dat, neighbor_obs.dat, site_obs.dat` are outputted. For details about each output file, see the file formats. The value of <Sz> can be extracted from the value following the `0 0` component of `site_obs.dat` or the `Local operator 0` in the standard output. By changing `G` in increments of 0.1 from 0 to 2.0 and running `tenes_simple` and `tenes`, the following result is obtained.

As an exapmle of the sample script, `tutorial_example.py` , `tutorial_read.py` are prepared in the `sample/01_transverse_field_ising` directory. The calculation will be done by typing the following command:

```
$ python tutorial_example.py
```

For MacBook2017 (1.4 GHz Intel Core i7), the calculation was finished in a few minutes. By typing the following command, G, energy, <Sz> and <Sx> are ouputted in the standard output:

```
$ python tutorial_read.py
```



As you can see from the figure, with increasing `G`, the <Sz> decreases gradually from `0.5` to `0`.

CHAPTER

FIVE

FILE FORMAT

5.1 Input file for `tense_simple`

- File format is TOML format.
- The input file has five sections : `model`, `parameter`, `lattice`, `observable`, `correlation` .
- In future, the file will be split by specifying a file name.

5.1.1 `parameter` section

The contents of the `parameter` section are copied directly to the `parameter` section of the input file of `tenes`. You can also specify the imaginary time step size for the imaginary time evolution operator in simple and full updates in the subsections `simple_update`, `full_update`.

Name	Description	Type	Default
`tau`	imaginary time step in the imaginary time evolution operator	Real	0.01

The following parameters are common to the `tenes` input file.

`parameter.tensor`

Name	Description	Type	Default
`D`	The virtual bond dimensions of the central tensor	Integer	2
`CHI`	The virtual bond dimensions of the angular transfer matrix	Integer	4
`save_dir`	Directory to write optimized tensors	Str	“”
`load_dir`	Directory to read initial tensor	Str	“”

- `save_dir` - Store optimized tensors below this directory. - When it is empty, the tensors are not saved.
- `load_dir` - Read various tensors from below this directory. - Must be the same degree of parallelism as when saved. - Not read if it is empty.

parameter.simple_update

Name	Description	Type	Default
num_step	Number of simple updates	Integer	0
lambda_cutoff	cutoff of the mean field to be considered zero in the simple update	Real	1e-12

parameter.full_update

Name	Description	Type	Default
num_step	Number of full updates	Integer	0
env_cutoff	Cutoff of singular values to be considered as zero when computing environment through full updates	Real	1e-12
inverse_precision	Cutoff of singular values to be considered as zero when computing the pseudoinverse matrix with full update	Real	1e-12
convergence_epsilon	Convergence criteria for truncation optimization with full update	Real	1e-12
iteration_max	Maximum iteration number for truncation optimization on full updates	Integer	1000
gauge_fix	Whether the tensor gauge is fixed	Boolean	true
fastfullupdate	Whether the Fast full update is adopted	Boolean	true

parameter.ctm

Name	Description	Type	Default
projector_cutoff	Cutoff of singular values to be considered as zero when computing CTM projectors	Real	1e-12
convergence_epsilon	CTM convergence criteria	Real	1e-10
iteration_max	Maximum iteration number of convergence for CTM	Integer	100
projector_corner	Whether to use only the 1/4 corner tensor in the CTM projector calculation	Boolean	true
use_rsvd	Whether to replace SVD with Random SVD	Boolean	false
rsvd_oversampling_factor	Ratio of the number of the oversampled elements to that of the obtained elements in the Random SVD method	Real	2.0

parameter.random

Name	Description	Type	Default
seed	Seed of the pseudo-random number generator used to initialize the tensor	Integer	11

Each MPI process has the own seed as `seed` plus the process ID (MPI rank).

Example

```
[parameter]
[parameter.tensor]
D  = 4     # tensor_dim
CHI  = 16  # env_dim

[parameter.simple_update]
num_step = 1000

[parameter.full_update]
num_step = 1

[parameter.ctm]
iteration_max = 5
```

5.1.2 `lattice` section

Specify the lattice information. Square lattice and honeycomb lattice are defined as lattice types.

Name	Description	Type
`type`	Lattice name (square lattice or honeycomb lattice)	Str
`L_sub`	Unit cell size	An integer or a list of two integers

When a list of two integers is passed as `L_sub`, the first element gives the value of `Lx` and the second one does `Ly`. If `L_sub` is an integer, Both `Lx` and `Ly` will have the same value. A list of three or more elements causes an error.

Sites in a unit cell are indexed starting from 0. These are arranged in order from the x direction.

Sites in a unit cell of `L_sub = [2,3]` are arranged as follows:

```
y
^     4 5
|     2 3
.->x  0 1
```

Square lattice

There are two types of bond, horizontal (0) and vertical (1) (corresponding to − and | in the below figure).

The unit cell for `L_sub = 2` is given as follows:

```
0   1
|   |
2 - 3 - 2
|   |
0 - 1 - 0
```

Honeycomb lattice

Unit cell size (Each element of `L_sub`) must be an even number.

There are 3 types of bonds: x, y, and z (corresponding to −, ~, | in the below figure). Each site with an even index has a rightward (x), a leftward (y), and an upward (z) bonds and each site with an odd index has a leftward (x), a rightward (y), and a bottomward (z) bonds.

The unit cell for `L_sub = 2` is given as follows:

```
0   1
    |
2 ~ 3 - 2
|
0 - 1 ~ 0
```

5.1.3 `model` section

Specify the type of the model. Only the Spin system can be spcified in ver. 0.1.

Name	Description	Type
`type`	The type of the model	Str

Spin system

Spin system

$$\mathcal{H} = \sum_{\langle ij \rangle} \left[\sum_{\alpha}^{x,y,z} J_{ij}^{\alpha} S_i^{\alpha} S_j^{\alpha} + B \left(\vec{S}_i \cdot \vec{S}_j \right)^2 \right] - \sum_i \left[h S_i^z + \Gamma S_i^x - D \left(S_i^z \right)^2 \right]$$

Table 1: :header: “Name”, “Description”, “Type”, “Default” :widths: 30, 30, 10, 10

`S`	Magnituide of the local spin	Real	0.5
`Jx`	The x component of the exchange interaction J	Real or a list of Real	1.0
`Jy`	The y component of the exchange interaction J	Real or a list of Real	1.0
`Jz`	The z component of the exchange interaction J	Real or a list of Real	1.0
`BQ`	Biquadratic interaction B	Real or a list of Real	0.0
`h`	longitudinal magnetic field h	Real	0.0
`G`	Transverse magnetic field Γ	Real	0.0
`D`	On-site spin anisotropy D	Real	0.0

By providing a list of exchange and biquadratic interactions, we can vary the magnitude of the interaction for each type of lattice bond. If the number of elements in the list is less than the type of lattice bond, the remainder is filled in with the last element specified.

5.1.4 `observable` section

By default, the local physical quantities used for physical quantities measurements: S^z and S^x . Measurements of more detailed physical quantities can be made by overwriting the format common to the input file of `tenes`. For details, See `observable` section *Input file for tenes*.

5.1.5 `correlation` section

For `tenes_simple` , correlation functions $C = \langle A(0)B(r)\rangle$ are not calculated by default. For calculating correlation functions, they have to be specified in the same file format as the input file of `tenes`. For details, See `correlation` section *Input file for tenes*.

5.2 Input file for `tenes`

- File format is TOML format.
- The input file has five sections: `parameter`, `lattice`, `evolution`, `observable`, `correlation`
  - In the future we will be able to split it by specifying the file name.

5.2.1 `parameter` section

Various parameters such as the number of updates are specified in this section. Only the parameters in this section have default values. This section has five sub sections, `tensor`, `simple_update`, `full_update`, `ctm`, `random`.

`parameter.tensor`

Name	Description	Type	Default
`D`	The virtual bond dimensions of the central tensor	Integer	2
`CHI`	The virtual bond dimensions of the angular transfer matrix	Integer	4
`save_dir`	Directory to write optimized tensors	Str	“”
`load_dir`	Directory to read initial tensor	Str	“”

- `save_dir` - Store optimized tensors below this directory. - When it is empty, the tensors are not saved.
- `load_dir` - Read various tensors from below this directory. - Must be same degree of parallelism as when saved. - Not read if it is empty.

`parameter.simple_update`

Name	Description	Type	Default
`num_step`	Number of simple updates	Integer	0
`lambda_cutoff`	cutoff of the mean field to be considered zero in the simple update	Real	1e-12

parameter.full_update

Name	Description	Type	Default
num_step	Number of full updates	Integer	0
env_cutoff	Cutoff of singular values to be considered as zero when computing environment through full updates	Real	1e-12
inverse_precision	Cutoff of singular values to be considered as zero when computing the pseudoinverse matrix with full update	Real	1e-12
convergence_epsilon	Convergence criteria for truncation optimization with full update	Real	1e-12
iteration_max	Maximum iteration number for truncation optimization on full updates	Integer	1000
gauge_fix	Whether the tensor gauge is fixed	Boolean	true
fastfullupdate	Whether the Fast full update is adopted	Boolean	true

parameter.ctm

Name	Description	Type	Default
projector_cutoff	Cutoff of singular values to be considered as zero when computing CTM projectors	Real	1e-12
convergence_epsilon	CTM convergence criteria	Real	1e-10
iteration_max	Maximum iteration number of convergence for CTM	Integer	100
projector_corner	Whether to use only the 1/4 corner tensor in the CTM projector calculation	Boolean	true
use_rsvd	Whether to replace SVD with Random SVD	Boolean	false
rsvd_oversampling_factor	Ratio of the number of the oversampled elements to that of the obtained elements in the Random SVD method	Real	2.0

parameter.random

Name	Description	Type	Default
seed	Seed of the pseudo-random number generator used to initialize the tensor	Integer	11

Each MPI process has the own seed as `seed` plus the process ID (MPI rank).

Example

```
[parameter]
[parameter.tensor]
D  = 4     # tensor_dim
```

(continues on next page)

(continued from previous page)

```
CHI  = 16  # env_dim

[parameter.simple_update]
num_step = 1000

[parameter.full_update]
num_step = 1

[parameter.ctm]
iteration_max = 5
```

5.2.2 `lattice` section

Specify the unit cell information. Unit cell has a shape of a rectangular with the size of `Lx` times `Ly`.

Name	Description	Type
`L_sub`	Unit cell size	An integer or a list of integer

When a list of two integers is passed as `L_sub`, the first element gives the value of `Lx` and the second one does `Ly`. If `L_sub` is an integer, Both `Lx` and `Ly` will have the same value. A list of three or more elements causes an error.

Sites in a unit cell are indexed starting from 0. These are arranged in order from the x direction.

Sites in a unit cell of `L_sub = [2,3]` are arranged as follows:

```
y
^     4 5
|     2 3
.->x  0 1
```

Information of bonds is given in the `evolution` and the `observable` sections.

5.2.3 `evolution` section

Define the imaginary time evolution opetrators used in simple and full updates.

Name	Description	Type
`matrix`	Matrix representation about the imaginary time evolution opetrators	A list of string
`simple_update`	The order of the bonds that act on the index of the imaginary time evolution operator in simple update	A list of string
`full_update`	The order of the bonds that act on the index of the imaginary time evolution operator in full update	A list of string

`matrix`

- One matrix is defined by a list of string.
- Columns are separated by one or more blanks, and rows are separated by one or more newlines.
- The order defined corresponds exactly to the number of the matrix. This order numbers are used to specify `simple_update` and `full_update` (0-origin).

`simple_update` and `full_update`

- One row represents one operator action.
- Each line consists of four fields: `int int string int`.
  1. A site to which bond connects
  2. A site to which bond connects
  3. Horizontal (h) or Vertical (v)
  4. Operator number (0-origin)

Example

```
[evolution]
simple_update = """
0 1 h 0
3 2 h 0
2 3 h 0
1 0 h 0
0 2 v 0
3 1 v 0
2 0 v 0
1 3 v 0
"""

full_update = """
0 1 h 0
3 2 h 0
2 3 h 0
1 0 h 0
0 2 v 0
3 1 v 0
2 0 v 0
1 3 v 0
"""

matrix = [
"""
0.9975031223974601 0.0 0.0 0.0
0.0 1.0025156589209967 -0.005012536523536887 0.0
0.0 -0.005012536523536888 1.0025156589209967 0.0
0.0 0.0 0.0 0.9975031223974601
"""
]
```

5.2.4 `observable` section

In this section, the information about physical quantities to be observed is specified.

Name	Description	Type
`local_operator`	Site opertor (ex. Sz)	A list of string
`hamiltonian`	Bond hamiltonian	A list of string
`hamiltonian_bonds`	Type of bond Hamiltonian and the set of bonds that act	string

`local_operator`, `hamiltonian`

Same as `evolution.matrix` . The order you define corresponds exactly to the index of the operator Hamiltonian.

`hamiltonian_bonds`

Same as `evolution.simple_update` .

Example

```
[observable]
local_operator = [
"""
  0.5  0.0
  0.0 -0.5
""",
"""
  0.0 0.5
  0.5 0.0
""",
]

hamiltonian_bonds = """
0 1 h 0
3 2 h 0
2 3 h 0
1 0 h 0
0 2 v 0
3 1 v 0
2 0 v 0
1 3 v 0
"""

hamiltonian = [
"""
  0.25   0.0    0.0     0.0
  0.0   -0.25   0.5     0.0
  0.0    0.5   -0.25    0.0
  0.0    0.0    0.0     0.25
""",
]
```

5.2.5 `correlation` section

In the following, the parameters about the correlation function $C = \langle A(0)B(r)\rangle$ is described.

Name	Description	Type
`r_max`	Maximum distance r of the correlation function	Integer
`operators`	Numbers of operators A and B that measure correlation functions	A list for Integer

The operators defined in the `observable` section are used.

Example

```
[correlation]
r_max = 5
operators = [[0,0], [0,1], [1,1]]
```

5.3 Output files

Output files are generated in the `output` directry.

5.3.1 `parameters.dat`

Paramters in the `parameter` and `lattice` sections defined in the input file are outputted.

5.3.2 `energy.dat`

The energy of each site is output.

5.3.3 `site_obs.dat`

- The expected values of the site operator are outputted.
- Each row consists of four columns.
  1. Index of the operator
  2. Index of the sites
  3. Real part of the expected value
  4. Imaginary part of the expected value

Example

```
# $1: op_index
# $2: site_index
# $3: real
# $4: imag

0 0 1.92549465249573365e-02 0.00000000000000000e+00
0 1 -1.92620814130195529e-02 0.00000000000000000e+00
0 2 -1.95243093055922252e-02 0.00000000000000000e+00
0 3 1.91619477632061150e-02 0.00000000000000000e+00
1 0 4.07206063348768799e-01 0.00000000000000000e+00
1 1 -4.07243511737157671e-01 0.00000000000000000e+00
1 2 -4.07255967738734126e-01 0.00000000000000000e+00
1 3 4.07308918791554009e-01 0.00000000000000000e+00
```

5.3.4 neighbor_obs.dat

- Nearest neighbor correlations for site operations are outputted.
- Each row consists of five columns.
  1. Index of the operator
  2. Index of the sites
  3. Index of the sites
  4. Real part of the expected value
  5. Imaginary part of the expected value

```
# $1: op_index
# $2: source_site
# $3: target_site
# $4: real
# $5: imag

0 0 1 -7.0592761506496890e-02 0.0000000000000000e+00
0 0 2 -7.2706845643005127e-02 0.0000000000000000e+00
0 1 0 -7.1328438595739229e-02 0.0000000000000000e+00
0 1 3 -7.1952334925611358e-02 0.0000000000000000e+00
0 2 3 -7.1261036489548304e-02 0.0000000000000000e+00
0 2 0 -7.1973150756101195e-02 0.0000000000000000e+00
0 3 2 -7.0563355823021007e-02 0.0000000000000000e+00
0 3 1 -7.2680375080734050e-02 0.0000000000000000e+00
1 0 1 -1.8594286923710335e-01 0.0000000000000000e+00
1 0 2 -1.8716473167754519e-01 0.0000000000000000e+00
1 1 0 -1.8636038255007659e-01 0.0000000000000000e+00
1 1 3 -1.8676845108636670e-01 0.0000000000000000e+00
1 2 3 -1.8638418190980594e-01 0.0000000000000000e+00
1 2 0 -1.8674757673269352e-01 0.0000000000000000e+00
1 3 2 -1.8597508952501360e-01 0.0000000000000000e+00
1 3 1 -1.8719652291687905e-01 0.0000000000000000e+00
```

5.3.5 correlation.dat

- Correlation functions are outputted.
- Each row consists of eight columns.
  1. Index of the left operator
  2. Site index of the left operator
  3. Index of the right operator
  4. Site index of the right operator
  5. Unit cell offset of the right operator (x)
  6. Unit cell offset of the right operator (y)
  7. Real part of the expected value
  8. Imaginary part of the expected value

Example

```
# $1: left_op
# $2: left_site
# $3: right_op
# $4: right_site
# $5: offset_x
# $6: offset_y
# $7: real
# $8: imag

0 0 0 1 0 0 -7.0592761506496792e-02 0.0000000000000000e+00
0 0 0 0 1 0 1.1966884322676102e-02 0.0000000000000000e+00
0 0 0 1 1 0 -2.4308622932000586e-03 0.0000000000000000e+00
0 0 0 0 2 0 7.4272919452849631e-04 0.0000000000000000e+00
0 0 0 1 2 0 -4.3879481941688542e-04 0.0000000000000000e+00
0 0 0 2 0 0 -7.2706845643005114e-02 0.0000000000000000e+00
0 0 0 0 0 1 1.2333984574662128e-02 0.0000000000000000e+00
0 0 0 2 0 1 -2.5011118624440735e-03 0.0000000000000000e+00
0 0 0 0 0 2 7.5460780658739152e-04 0.0000000000000000e+00
0 0 0 2 0 2 -4.4773455996967955e-04 0.0000000000000000e+00
1 0 1 1 0 0 -1.8594286923710324e-01 0.0000000000000000e+00
...
1 3 1 1 0 3 -1.6587424589146155e-01 0.0000000000000000e+00
```

5.3.6 `time.dat`

The calculation time is outputted.

CHAPTER

SIX

REFERENCES

Please see the following documents if you want to know the algorithms used in `TeNeS`.

- Lecture slides and video
  - Slide: https://www.issp.u-tokyo.ac.jp/public/caqmp2019/slides/808L_Okubo.pdf
  - youtube: https://www.youtube.com/watch?v=EL9iGwoqZes&list=PLyL_XwLqTTm6roebw6MgIK0ljrN8cSQUX
- About Simple update
  - H. C. Jiang, Z. Y. Weng, and T. Xiang, Phys. Rev. Lett. **101**, 090603 (2008).
- About Full update (including the evaluation of the CTMRG's environment)
  - J. Jordan, R. Orús, G. Vidal, F. Verstraete, and J. I. Cirac, Phys. Rev. Lett. **101**, 250602 (2008).
  - R. Orús and G. Vidal, Phys. Rev. B **80**, 094403 (2009).
  - P. Corboz, T. M. Rice, and M. Troyer, Phys. Rev. Lett. **113**, 046402 (2014).
- About Fast Full update, Gauge fixing
  - Ho N. Phien, Johann A. Bengua, Hoang D. Tuan, Philippe Corboz, and Román Orús Phys. Rev. B **92**, 035142 (2015).
- Reviews
  - Román Orús, Annals of Physics **349**,117 (2014).
  - Román Orús, Nature Reviews Physics **1**, 538 (2019).

CHAPTER

SEVEN

ACKNOWLEDGEMENT

`TeNeS` was supported by MEXT as "Exploratory Challenge on Post-K computer" (Frontiers of Basic Science: Challenging the Limits) and "Priority Issue on Post-K computer" (Creation of New Functional Devices and High-Performance Materials to Support Next-Generation Industries). We also would also like to express our thanks for the support of the "*Project for advancement of software usability in materials science*" of The Institute for Solid State Physics, The University of Tokyo, for the development of `TeNeS`.

CHAPTER

EIGHT

CONTACTS

- About Bugs

  Please report all problems and bugs on the github Issues page

  To resolve bugs early, follow these guidelines when reporting:

  - Please specify the version of `TeNeS` you are using.
  - If there are problems for installation, please inform us about your operating system and the compiler, and include the input / output and CMakeCache.txt when you run cmake and make.
  - If a problem occurs during execution, enter the input file used for execution and its output.

  Thank you for your cooperation.

- Others

  If you have any questions about topics related to your research that are difficult to consult at Issues on GitHub, please contact the following contact information.

  E-mail: `tenes-dev__at__issp.u-tokyo.ac.jp` (replace _at_ by @)