User guide

$\mathcal{H}\Phi$ version 1.2.0

Nov. 15, 2016

Contents

1
What is $\mathcal{H}\Phi$?

1.1 What is $\mathcal{H}\Phi$?

Comparison between experimental observation and theoretical analysis is a crucial step in condensed-matter physics research. Temperature dependence of specific heat and magnetic susceptibility, for example, have been studied to extract nature of low energy excitations of and magnetic interactions among electrons, respectively, through comparison with theories such as Landau's Fermi liquid theory and Curie-Weiss law.

For the flexible and quantitative comparison with experimental data, an exact diagonalization approach [1] is one of the most reliable numerical tools without any approximation or inspiration of genius. For last few decades, a numerical diagonalization package for quantum spin hamiltonians, TITPACK developed by Prof. Hidetoshi Nishimori in Tokyo Institute of Technology, has been widely used in the condensed-matter physics community. Nevertheless, limitation of computational resources had hindered the non-expert users from applying the package to quantum systems with large number of electrons or spins.

In contrast, recent and rapid development of parallel computing infrastructure opens up new avenues for user-friendly larger scale diagonalizations up to 18 site Hubbard clusters or 36 S=1/2 quantum spins. In addition, recent advances in quantum statistical mechanics [2–5] enable us to calculate finite temperature properties of quantum many-body systems with computational costs similar to calculations of ground state properties, which also enables us to compare theoretical results for temperature dependence of, for example, specific heat and magnetic susceptibility with experimental results quantitatively [6]. To utilize the parallel computing infrastructure with narrow bandwidth and distributed-memory architectures, efficient, user-friendly, and highly parallelized diagonalization packages are highly desirable.

$\mathcal{H}\Phi$, a flexible diagonalization package for solving quantum lattice hamiltonians, has been developed to be such a descendant of the pioneering package TITPACK. The Lanczos method for calculations of the ground state and a few excited states properties, and finite temperature calculations based on thermal pure quantum states [5] are implemented in the package $\mathcal{H}\Phi$, with an easy-to-use and flexible user interface. By using $\mathcal{H}\Phi$, you can analyze a wide range of quantum lattice hamiltonians including simple Hubbard and Heisenberg models, multi-band extensions of the Hubbard model, exchange couplings that break SU(2) symmetry of quantum spins such as Dzyaloshinskii-Moriya and Kitaev interactions, and Kondo lattice models describing itinerant electrons coupled with quantum spins. $\mathcal{H}\Phi$ calculates a variety of physical quantities such as internal energy at zero temperature or finite temperatures, temperature dependence of specific heat, charge/spin structure factors, and so on. A broad spectrum of users including experimental scientists is cordially welcome.

1.1.1 License

The distribution of the program package and the source codes for $\mathcal{H}\Phi$ follow GNU General Public License version 3 (GPL v3).

1.1.2 Copyright

1.1.3 Contributors

This software is developed by following contributors.

- ver.1.2 (released at 2016/11/14)
- ver.1.1 (released at 2016/5/13)
- ver.1.0 (released at 2016/4/5)
  - **Developers**
    - Takahiro Misawa
      (Department of Applied Physics, The University of Tokyo)
    - Kazuyoshi Yoshimi
      (The Institute for Solid State Physics, The University of Tokyo)
    - Mitsuaki Kawamura
      (The Institute for Solid State Physics, The University of Tokyo)
    - Youhei Yamaji
      (Department of Applied Physics, The University of Tokyo)
    - Synge Todo
      (Department of Physics, The University of Tokyo)
  - **Project coordinator**
    - Naoki Kawashima
      (The Institute for Solid State Physics, The University of Tokyo)
- ver.0.3 (released at 2016/2/24)
- ver.0.2 (released at 2015/12/28)
- ver.0.1 (released at 2015/10/09)

- **Developers**
  - Takahiro Misawa
    (Department of Applied Physics, The University of Tokyo)
  - Kazuyoshi Yoshimi
    (The Institute for Solid State Physics, The University of Tokyo)
  - Mitsuaki Kawamura
    (The Institute for Solid State Physics, The University of Tokyo)
- **Advisers**
  - Youhei Yamaji
    (Department of Applied Physics, The University of Tokyo)
  - Synge Todo
    (Department of Physics, The University of Tokyo)
- **Project coordinator**
  - Naoki Kawashima
    (The Institute for Solid State Physics, The University of Tokyo)

1.2 Operating environment

$\mathcal{H}\Phi$ is tested in the following platforms:

- The supercomputer system-B “sekirei” and system-C “maki” in ISSP
- Linux PC + intel compiler
- Linux PC + gcc
- Mac + gcc

2
How to use $\mathcal{H}\Phi$?

2.1 Prerequisite

$\mathcal{H}\Phi$ requires the following packages:

- C compiler (intel, Fujitsu, GNU, etc.)
- LAPACK library (intel MKL, Fujitsu, ATLAS, etc.)
- MPI library (If you do not use MPI, it is not necessary.)

Tips

E. g. / Settings of intel compiler

When you use the intel compiler, you can use easily scripts attached to the compiler. In the case of the bash in 64 bit OS, write the following in your `~/.bashrc`:

```
source /opt/intel/bin/compilervars.sh intel64
```

or

```
source /opt/intel/bin/iccvars.sh intel64
source /opt/intel/mkl/bin/mklvars.sh
```

Please read manuals of your compiler/library for more information.

2.2 Installation

You can download $\mathcal{H}\Phi$ in the following place.
`https://github.com/QLMS/HPhi/releases`

You can obtain the $\mathcal{H}\Phi$ directory by typing

```
$ tar xzvf HPhi-xxx.tar.gz
```

There are two kind of procedures to install $\mathcal{H}\Phi$.

2.2.1 Using `HPhiconfig.sh`

Please run `HPhiconfig.sh` script in the $\mathcal{H}\Phi$ directory as follow (for ISSP system-B "sekirei"):

```
$ bash HPhiconfig.sh sekirei
```

Then environmental configuration file `make.sys` is generated in `src/` directory. The command-line argument of `HPhiconfig.sh` is as follows:

- `sekirei` : ISSP system-B "sekirei"
- `maki` : ISSP system-C "maki"
- `sr` : HITACHI SR16000
- `intel` : intel compiler + Linux PC
- `mpicc-intel` : intel compiler + MPI (excepting intelMPI) + Linux PC
- `gcc` : GCC + Linux PC
- `gcc-mac` : GCC + Mac

`make.sys` is as follows (for ISSP-system-B "sekirei"):

```
CC = mpiicc
LAPACK_FLAGS = -Dlapack -mkl=parallel
FLAGS = -qopenmp  -O3 -xCORE-AVX2 -mcmodel=large -shared-intel -D MPI
MTFLAGS = -DDSFMT_MEXP=19937 $(FLAGS)
INCLUDE_DIR=./include
```

We explain macros of this file as:

- `CC` : The Compilation command (`icc`, `gcc`, `fccpx`)
- `LAPACK_FLAGS` : Compilation options for LAPACK. `-Dlapack` can not be removed.
- `FLAGS` : Other compilation options. OpenMP utilization option (`-openmp`, `-fopenmp`, `-qopenmp`, etc.) must be specified. When you use MPI, please set `-D MPI`.
- `MTFLAGS`, `INCLUDE_DIR` : Options for the Mersenne Twister and additional include directory. You do not have to modify them.

Then you are ready to compile HPhi. Please type

```
$ make HPhi
```

and obtain an executable `HPhi` in `src/` directory; you should add this directory to the `$PATH`.

Tips

You can make a PATH to $\mathcal{H}\Phi$ as follows:

`$ export PATH=${PATH}:`*HPhi_top_directory*`/src/`

If you keep this PATH, you should write above in `~/.bashrc` (for `bash` as a login shell)

2.2.2 Using `cmake`

Tips

Before using cmake for sekirei, you must type

```
source /home/issp/materiapps/tool/env.sh
```

while for maki, you must type

```
source /global/app/materiapps/tool/env.sh
```

We can compile Hphi as

```
cd $HOME/build/hphi
cmake -DCONFIG=gcc $PathTohphi
make
```

Here, we set a path to $\mathcal{H}\Phi$ as `$PathTohphi` and to a build directory as `$HOME/build/hphi`. After compiling, a src folder is constructed below a `$HOME/build/hphi` folder and obtain an executable `HPhi` in `src/` directory. When there is not a MPI library in your system, an executable `HPhi` is automatically compiled without a MPI library.

In the above example, we compile $\mathcal{H}\Phi$ by using a gcc compiler. We can select a compiler by using following options

- `sekirei` : ISSP system-B "sekirei"
- `fujitsu` : Fujitsu compiler (ISSP system-C "maki")
- `intel` : intel compiler + Linux PC
- `gcc` : GCC compiler + Linux PC.

An example for compiling $\mathcal{H}\Phi$ by an intel compiler is shown as follows,

```
mkdir ./build
cd ./build
cmake -DCONFIG=intel ../
make
```

After compiling, a `src` folder is made below the `build` folder and an execute $\mathcal{H}\Phi$ is made in the `src` folder. It is noted that we must delete the `build` folder and do the above works again when we change the compilers.

2.3 Directory structure

When HPhi-xxx.tar.gz is unzipped, the following directory structure is composed.

```
|--CMakeLists.txt
|--COPYING
|--config/
|     |--fujitsu.cmake
|     |--gcc.cmake
|     |--intel.cmake
|     ---sekirei.cmake
|--doc/
|     |--en/
|     |--jp/
|     |--userguide_en.pdf
|     ---userguide_jp.pdf
|--HPhiconfig.sh
|--samples/
|     |--Expert/
|     |     |--Hubbard/
|     |     |     |-square/
|     |     |     |       |--*.def
|     |     |     |       |--output_FullDiag/*.dat
|     |     |     |       |--output_Lanczos/*.dat
|     |     |     |       ---output_TPQ/*.dat
|     |     |     --triangular/...
|     |     |--Kondo/chain/...
|     |     ---Spin/
|     |           |-HeisenbergChain/...
|     |           |-HeisenbergSquare/...
|     |           |-Kagome/...
|     |           |-Kagome36Boost/...
|     |           |-Kitaev/...
|     |           --S1Chain/...
|     ---Standard/
|           |--Hubbard/
|           |     |-square/
|           |     |   |--StdFace.def
|           |     |   |--output_FullDiag/*.dat
|           |     |   |--output_Lanczos/*.dat
|          |    |   ---output_TPQ/*.dat
|          |    --triangular/...
|          |--Kondo/chain/...
|          ---Spin/
|               |-HeisenbergChain/...
|               |-HeisenbergSquare/...
|               |-Kitaev/...
|               --S1Chain/...
|--src/
|    |--*.c
|    |--CMakeLists.txt
|    |--include/*.h
|    |--makefile_src
|    ---StdFace/
|--test/
---test_tool/
```

2.4 Basic usage

$\mathcal{H}\Phi$ has two modes; standard mode and expert mode. Here, the basic flows of calculations by standard and expert modes are shown.

2.4.1 *Standard* mode

The procedure of calculation through the standard mode is shown as follows:

1. Make a directory for a calculation scenario.

   First, you make a working directory for the calculation.

2. Make input files for standard mode

   In the standard mode, you can choose a model (the Heisenberg model, the Hubbard model, etc.) and a lattice (the square lattice, the triangular lattice, etc.) from ones provided; you can specify some parameters (such as the first/second nearest neighbor hopping integrals, the on-site Coulomb integral, etc.) for them. Finally, you have to specify the numerical method (such as the Lanczos method) employed in this calculation. The input file format is described in the Sec. 4.

3. Run

   Run a executable `HPhi` in terminal by setting option "`-s`" (or "`--standard`") and the name of input file written in previous step.

   - Serial/OpenMP parallel

     `$` *Path*`/HPhi -s` *Input_file_name*

- MPI parallel/ Hybrid parallel

  `$ mpiexec -np` *number_of_processes* *Path*`/HPhi -s` *Input_file_name*

  When you use a queuing system in workstations or super computers, sometimes the number of processes is specified as an argument for the job-submitting command. If you need more information, please refer manuals for your system. A number of processes depend on a target of system for models. The details of setting a number of processes are shown in 2.4.3.

4. Watch calculation logs

   Log files are outputted in the "output" folder which is automatically made in the directory for a calculation scenario. The details of output files are shown in 4.3.

5. Results

   If the calculation is finished normally, the result files are outputted in the "output" folder. The details of output files are shown in 4.3.

Tips

The number of threads for OpenMP

If you specify the number of OpenMP threads for $\mathcal{H}\Phi$, you should set it as follows (in case of 16 threads) before the running:

```
export OMP_NUM_THREADS=16
```

2.4.2 *Expert* mode

The procedure of calculation for expert mode is shown as follows.

1. Make a directory for a calculation scenario.
   First, you make a directory named as a calculation scenario (you can attach an arbitrary name to a directory).

2. Make input files for expert mode
   For expert mode, you should make input files for constructing Hamiltonian operators, calculation condition and a list file for the filenames of input files (see the file formats shown in 4.2).
   Note: A List file can be easily made by using standard mode.

3. Run
   Run $\mathcal{H}\Phi$ in terminal by setting option "`-e`" (or "`--expert`") and a file name of a list file.

- Serial/OpenMP

  `$` *Path*`/HPhi -e` *Input_List_file_name*

- MPI/Hybrid

  `$ mpiexec -np` *number_of_processes* *Path*`/HPhi -e` *Input_List_file_name*
  A number of processes depend on a target of system for models. The details of setting a number of processes are shown in 2.4.3.

4. Under running
   Log files are outputted in the "output" folder which is automatically made in the directory for a calculation scenario. The details of output files are shown in 4.3.

5. Results
   If the calculation is finished normally, the result files are outputted in the "output" folder. The details of output files are shown in 4.3.

2.4.3 Setting a process number for MPI/Hybrid parallelization

For using MPI/Hybrid parallelization, a process number must be set as follows.

1. Standard mode

   - Hubbard / Kondo model
     When `model` in an input file for Standard mode is set as `"Fermion Hubbard"`, `"Kondo Lattice"` or `"Fermion HubbardGC"`, a process number must be equal to 4^n.
   - Spin model
     When `model` in an input file for Standard mode is set as `"Spin"` or `"SpinGC"`, a process number must be equal to (`2S`+1)^n where `2S` is set in the input file (a default value is 1).

2. Expert mode

   - Hubbard / Kondo model
     When a model is selected as fermion Hubbard model or Kondo model by setting `CalcModel` in a **CalcMod** file, a process number must be equal to 4^n. See 4.2.2 for details of a `CalcModel` file.
   - Spin model
     When a model is selected as Spin model by setting `CalcModel` in a **CalcMod** file, a process number is fixed by a **LocSpin** file. A process number must be equal to a number calculated by multiplying a state number of localized spin (`2S`+1) in descending order about site numbers. See 4.2.4 for details of a **LocSpin** file.
     For example, when a **LocSpin** file is given as follws, a process number must be equal to $2 = 1 + 1,\ 6 = 2 \times (2 + 1),\ 24 = 6 \times (3 + 1)$.

```
================================
NlocalSpin     3
================================
========i_0IteElc_2S ======
================================
    0      3
    1      2
    2      1
```

2.4.4 Printing version ID

By using `-v` option as follows, you can check which version of $\mathcal{H}\Phi$ you are using.

```
$ PATH/HPhi -v
```

3
Tutorial

3.1 *Standard* mode

3.1.1 Heisenberg model

This tutorial should be performed in

```
samples/Standard/Spin/HeisenbergChain/
```

The input file, reference outputs, and the redirected standard output are provided as follows:

```
samples/Standard/Spin/HeisenbergChain/StdFace.def
samples/Standard/Spin/HeisenbergChain/output_FullDiag/
samples/Standard/Spin/HeisenbergChain/output_Lanczos/
samples/Standard/Spin/HeisenbergChain/output_TPQ/
samples/Standard/Spin/HeisenbergChain/FullDiag.out
samples/Standard/Spin/HeisenbergChain/Lanczos.out
samples/Standard/Spin/HeisenbergChain/TPQ.out
```

In this case, we treat the one dimensional antiferromagnetic Heisenberg chain which has a nearest neighbor spin coupling.

$$\hat{H} = J \sum_{i=1}^{N_{\rm site}} \hat{\boldsymbol{S}}_i \cdot \hat{\boldsymbol{S}}_{i+1} \tag{3.1}$$

The input file is as follows:

```
L = 16
model = "Spin"
method = "Lanczos"
lattice = "chain lattice"
J = 1.0
2Sz = 0
```

In this tutorial, J and the number of sites are set to 1 (arbitral unit) and 16 respectively.

Log output

Log messages are outputted to the standard output. Log files for calculation procedure are made in "output" directory which is automatically created. In this example, the following files are outputted.

```
CHECK_InterAll.dat      Time_CG_EigenVector.dat  zvo_Lanczos_Step.dat
CHECK_Memory.dat        WarningOnTransfer.dat    zvo_sz_TimeKeeper.dat
CHECK_Sdim.dat          zvo_TimeKeeper.dat
```

The details for the outputted files are shown in 4.3.1-4.3.11.
We execute
`$ Path/HPhi -s StdFace.def`
and obtain following standard outputs (a compilation mode is MPI parallel/ Hybrid parallel):

```
#####  Parallelization Info.  #####

  OpenMP threads : 4
  MPI PEs : 1


######  Standard Intarface Mode STARTS  ######

  Open Standard-Mode Inputfile ./StdFace.def

  KEYWORD : l                    | VALUE : 16
  KEYWORD : model                | VALUE : spin
  KEYWORD : method               | VALUE : lanczos
  Skipping a line.
  Skipping a line.
  KEYWORD : lattice              | VALUE : chainlattice
  KEYWORD : j                    | VALUE : 1.0
  KEYWORD : 2sz                  | VALUE : 0

#######  Parameter Summary  #######

                L = 16
                a = 1.00000     ######  DEFAULT VALUE IS USED  ######
               2S = 1           ######  DEFAULT VALUE IS USED  ######
                h = 0.00000     ######  DEFAULT VALUE IS USED  ######
            Gamma = 0.00000     ######  DEFAULT VALUE IS USED  ######
                D = 0.00000     ######  DEFAULT VALUE IS USED  ######
                J = 1.00000
               Jz = 1.00000     ######  DEFAULT VALUE IS USED  ######
              Jxy = 1.00000     ######  DEFAULT VALUE IS USED  ######
               Jx = 1.00000     ######  DEFAULT VALUE IS USED  ######
               Jy = 1.00000     ######  DEFAULT VALUE IS USED  ######
               J' = 0.00000     ######  DEFAULT VALUE IS USED  ######
```

```
             Jz' = 0.00000     ######  DEFAULT VALUE IS USED  ######
            Jxy' = 0.00000     ######  DEFAULT VALUE IS USED  ######
             Jx' = 0.00000     ######  DEFAULT VALUE IS USED  ######
             Jy' = 0.00000     ######  DEFAULT VALUE IS USED  ######
      LargeValue = 1           ######  DEFAULT VALUE IS USED  ######

        filehead = zvo         ######  DEFAULT VALUE IS USED  ######
     Lanczos_max = 2000        ######  DEFAULT VALUE IS USED  ######
      initial_iv = 1           ######  DEFAULT VALUE IS USED  ######
            nvec = 1           ######  DEFAULT VALUE IS USED  ######
            exct = 1           ######  DEFAULT VALUE IS USED  ######
      LanczosEps = 14          ######  DEFAULT VALUE IS USED  ######
   LanczosTarget = 2           ######  DEFAULT VALUE IS USED  ######
          NumAve = 5           ######  DEFAULT VALUE IS USED  ######
    ExpecInterval = 20          ######  DEFAULT VALUE IS USED  ######
             2Sz = 0
     ioutputmode = 1           ######  DEFAULT VALUE IS USED  ######

######  Print Expert input files  ######

    zlocspin.def is written.
      zTrans.def is written.
   zInterAll.def is written.
    namelist.def is written.
     calcmod.def is written.
     modpara.def is written.
    greenone.def is written.
    greentwo.def is written.

######  Input files are generated.  ######

  Read File 'namelist.def'.
  Read File 'calcmod.def' for CalcMod.
  Read File 'modpara.def' for ModPara.
  Read File 'zlocspn.def' for LocSpin.
  Read File 'zTrans.def' for Trans.
  Read File 'zInterAll.def' for InterAll.
  Read File 'greenone.def' for OneBodyG.
  Read File 'greentwo.def' for TwoBodyG.

######  Definition files are correct.  ######

  Read File 'zlocspn.def'.
  Read File 'zTrans.def'.
  Read File 'zInterAll.def'.
  Read File 'greenone.def'.
  Read File 'greentwo.def'.
```

```
######  Indices and Parameters of Definition files(*.def) are complete.  ######

  MAX DIMENSION idim_max=12870
  APPROXIMATE REQUIRED MEMORY  max_mem=0.001236 GB


######  MPI site separation summary  ######

  INTRA process site
    Site    Bit
       0       2
       1       2
       2       2
       3       2
       4       2
       5       2
       6       2
       7       2
       8       2
       9       2
      10       2
      11       2
      12       2
      13       2
      14       2
      15       2

  INTER process site
    Site    Bit

  Process element info
    Process       Dimension   Nup  Ndown  Nelec  Total2Sz   State
          0           12870     8      8      8         0

   Total dimension : 12870


######  LARGE ALLOCATE FINISH !  ######

  Start: Calculate HilbertNum for fixed Sz.
  End  : Calculate HilbertNum for fixed Sz.

  Start: Calculate diagaonal components of Hamiltonian.
  End  : Calculate diagaonal components of Hamiltonian.

######  Start: Calculate Lanczos Eigenvalue.  ######

  initial_mode=0 normal: iv = 6437 i_max=12870 k_exct =1
  LanczosStep  E[1] E[2] E[3] E[4], E_Max / Nsite
  stp = 2 0.7192235936 2.7807764064 xxxxxxxxxx xxxxxxxxx
  stp = 4 -1.1878294823 0.6833997592 2.1864630296 3.4242789858 0.2140174366
  stp = 6 -2.4623912732 -0.8925857144 0.5104359160 1.7443963706 0.2277205490
  stp = 8 -3.5878334190 -2.0969913075 -0.8250723080 0.3369317092 0.2325450330
```

(snip)

```
  stp = 60 -7.1422963606 -6.8721066784 -6.6965474265 -6.5234070570 0.2499999989
  stp = 62 -7.1422963606 -6.8721066784 -6.6965474266 -6.5234070573 0.2499999996
  stp = 64 -7.1422963606 -6.8721066784 -6.6965474266 -6.5234070574 0.2499999999

######  End  : Calculate Lanczos EigenValue.  ######

######  Start: Calculate Eigenvector.  ######

  Start: Calculate Lanczos Eigenvector.
  End  : Calculate Lanczos Eigenvector.
  Start: Calculate Energy.
  End  : Calculate Energy.

  Accuracy check !!!
  LanczosEnergy = -7.14229636061676e+00
  EnergyByVec   = -7.14229636061616e+00
  diff_ene      = 8.50586433832080e-14
  var           = 9.97303843188625e-14
  Accuracy of Lanczos vectors is enough.

######  End  : Calculate Eigenvector.  ######

  Start: Calculate one body Green functions.
  End  : Calculate one body Green functions.

  Start: Calculate two bodies Green functions.
  End  : Calculate two bodies Green functions.
```

In the beginning of this run, files describing the detail of considered Hamiltonian (`zlocspin.def`, `zTrans.def`, `zInterAll.def`, `namelist.def`, `calcmod.def`, `modpara.def`) and files specifying elements of correlation functions that will be calculated (`greenone.def`, `greentwo.def`) are generated.

Tips

In the standard outputs at each even step, the quantity of `E_Max / Nsite` column indicates the maximum energy per sites (In this example, it converges to `0.2499999999=0.25`). In the TPQ calculation, you can set `Largevalue` larger than `E_Max / Nsite`.

Outputs for calculation results

Lanczos method

When a calculation by Lanczos method is finished normally, eigenenergies, one-body Green's functions and two-body Green's functions are calculated and outputted to the files, respectively. In this sample, following files are outputted.

```
zvo_energy.dat zvo_cisajs.dat
zvo_cisajscktalt.dat
```

For standard mode, all pairs of $\langle n_{i\sigma} \rangle$ are calculated as one-body Green's functions and those of $\langle n_{i\sigma} n_{j\sigma'} \rangle$ are calculated as two-body Green's functions on the basis of the definition files, `greenone.def` and `greentwo.def`.
When accuracy of Lanczos vectors is enough, one-body and two-body Green's functions are calculated by eigenvectors obtained by Lanczos method. While accuracy of Lanczos vectors is *not* enough, a message "Accuracy of Lanczos vector is not enough" is outputted to the standard output and one-body and two-body Green's functions are calculated by eigenvectors obtained by CG method. The details of output files are shown in 4.3.12, 4.3.20, 4.3.21.

TPQ method

When `method="TPQ"` is selected in an input file, a calculation by TPQ method is started. After finishing of the calculation normally, following files are outputted, where %% is a number of run and && is a number of steps for TPQ method.

```
Norm_rand%%.dat SS_rand%%.dat
zvo_cisajs_set%%step&&.dat
zvo_cisajscktalt_set%%step&&.dat
```

In Norm_rand%%.dat, basic informations such as inverse of temperature and a norm of wave function before normalization are outputted with a TPQ step for each number of runs. In SS_rand%%.dat, physical quantities such as inverse of temperature, energy and expected value of square of Hamiltonian are outputted with a TPQ step for each number of runs. In zvo_cisajs_set%%step&&.dat and zvo_cisajscktalt_set%%step&&.dat, one-body and two-body Green's functions are outputted for each number of a TPQ steps and runs. The details of these files are shown in 4.3.15, 4.3.16, 4.3.20, 4.3.21.

Full diagonalization method

When `method = "fulldiag"` is selected in an input file, a calculation by Full diagonalization method is started. After finishing of the calculation normally, following files are outputted, where xx is a number of an eigenstate counting from 0.

```
Eigenvalue.dat zvo_cisajs_eigen_xx.dat
zvo_cisajscktalt_eigen_xx.dat  zvo_phys_Nup4_Ndown4.dat
```

In Eigenvalue.dat, an eigennumber and an eigenvalue are outputted for each lines. In zvo_cisajs_eigen_xx.dat and zvo_cisajscktalt_eigen_xx.dat, one-body Green's functions and two-body Green's functions are outputted for each eigennumber. In zvo_phys_Nup4_Ndown4.dat, physical quantities such as expected values of energy and doublon are outputted. The details of these files are shown in 4.3.18 - 4.3.21.

3.1.2 Other tutorials

There are following tutorials in `samples/Standard/`.

- The Hubbard model on the two dimensional square lattice (`samples/Standard/Hubbard/square/`)
- The Hubbard model on the two dimensional triangular lattice (`samples/Standard/Hubbard/triangular/`)
- The one dimensional Kondo chain (`samples/Standard/Kondo/chain/`)
- The one dimensional antiferromagnetic Heisenberg chain (`samples/Standard/Spin/HeisenbergChain/HeisenbergChain/`)
- The antiferromagnetic Heisenberg model on the two dimensional square lattice (`samples/Standard/Spin/HeisenbergSquare/`)
- The Kitaev model with 2×3 unit cells to Honeycomb lattice (`samples/Standard/Spin/Kitaev/`)

We can perform these tutorials in the same way to the previous one.

3.2 *Expert* mode

For expert mode, following input files are needed

1. A file list for input files,
2. Files for basic parameters,
3. Files for constructing Hamiltonian,
4. Files for setting output components.

The process after calculation is same as standard mode. In this section, we show the demonstration for one dimensional antiferromagnetic Heisenberg chain model which has a nearest neighbor spin coupling,

$$H = \sum_{i=0}^{15} J\boldsymbol{S}_i \cdot \boldsymbol{S}_{i+1}, \tag{3.2}$$

where $J = 2$, $\boldsymbol{S}_{16} = \boldsymbol{S}_0$.
We use following input files in samples/Expert/Spin/HeisenbergChain.

```
calcmod.def   greentwo.def  namelist.def  zTrans.def
greenone.def  modpara.def   zInterAll.def zlocspn.def
```

3.2.1 A file list for input files

In namelist.def, kinds of input files and filenames are defined as shown below. By writing keyword and filenames at each lines, kinds of files are distinguished. The details for namelist.def are shown in 4.2.1.

```
CalcMod calcmod.def
ModPara modpara.def
LocSpin zlocspn.def
Trans zTrans.def
InterAll zInterAll.def
OneBodyG greenone.def
TwoBodyG greentwo.def
```

3.2.2 Files for basic parameters

In this subsection, we show the way to set a calculation mode, parameters for calculation and positions of localized spins.

Setting a calculation mode

Calculation mode is set in a CalcMod file (in this sample file, calcmod.def). The contents of files are shown as below.

```
#CalcType = 0:Lanczos, 1:TPQCalc, 2:FullDiag
#CalcMod = 0:Hubbard, 1:Spin, 2:Kondo, 3:HubbardGC,
4:SpinGC, 5:KondoGC
CalcType   0
CalcModel   1
```

We select a calculation method by CalcType and a target model by CalcModel. In this sample, we set Lanczos method as a calculation method and a target model as a spin system (canonical ensemble). The details of a CalcMod file are shown in 4.2.2.

Setting parameters for calculation

Parameter for calculation are set in a ModPara file (in this sample, modpara.def). The contents of files are shown as below.

```
--------------------
Model_Parameters   0
--------------------
VMC_Cal_Parameters
--------------------
CDataFileHead  zvo
CParaFileHead  zqp
--------------------
Nsite          16
Ncond          0
2Sz            0
Lanczos_max    1000
initial_iv     12
nvec           1
exct           1
LanczosEps     14
LanczosTarget  2
LargeValue     12
NumAve         5
ExpecInterval  20
```

In this file, we set parameters for calculation such as a site number, a total number of conduction electrons, a total S_z and a number of Lanczos step etc. The details of ModPara file are shown in 4.2.3.

Setting positions of localized spins

Positions and S of localized spins are defined by a LocSpin file (in this sample, locspn.def). The contents of files are shown as below.

```
================================
NlocalSpin    16
================================
========i_0LocSpn_1IteElc ======
================================
    0      1
    1      1
    2      1
    3      1
    4      1
    5      1
...
```

When CalcModel in a CalcMod file is set as spin system, all sites automatically treated as localized spins. The details of a LocSpin file is shown in 4.2.4.

3.2.3 Files for constructing Hamiltonian

After setting basic parameters, we make input files for constructing Hamiltonian. Since calculations are done by the representation of fermion operators in $\mathcal{H}\Phi$, we must rewrite spin operator. For example, in the case of $S = 1/2$, we rewrite the equation by using the following relation

$$S_z^{(i)} = (c_{i\uparrow}^\dagger c_{i\uparrow} - c_{i\downarrow}^\dagger c_{i\downarrow})/2, \tag{3.3}$$
$$S_+^{(i)} = c_{i\uparrow}^\dagger c_{i\downarrow}, \tag{3.4}$$
$$S_-^{(i)} = c_{i\downarrow}^\dagger c_{i\uparrow}. \tag{3.5}$$

Setting transfer integrals

In a Trans file (in this sample, zTrans.def), we set a transfer part of Hamiltonian,

$$H+ = -\sum_{ij\sigma_1\sigma2} t_{ij\sigma_1\sigma2} c_{i\sigma_1}^\dagger c_{j\sigma_2}. \tag{3.6}$$

The contents of files are shown as below.

```
========================
NTransfer      0
========================
========i_j_s_tijs======
========================
```

We can use this term when an electric magnetic field is added in spin system. For example, when an magnetic field is added at a site 1 such as $-0.5S_z^{(1)}$ for $S = 1/2$, this term can be rewritten as $-0.5/2(c_{1\uparrow}^\dagger c_{1\uparrow} - c_{1\downarrow}^\dagger c_{1\downarrow})$. Thus, the input file becomes as follows.

```
========================
NTransfer      1
========================
========i_j_s_tijs======
========================
1 0 1 0 -0.25 0
1 1 1 1 0.25 0
```

The details for a Trans file are shown in 4.2.5.

Setting general two-body interactions

In an InterAll file (in this sample, zInterall.def), we set a general two-body interaction part of Hamiltonian,

$$H+ = \sum_{i,j,k,l} \sum_{\sigma_1,\sigma_2,\sigma_3,\sigma_4} I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c^{\dagger}_{i\sigma_1} c_{j\sigma_2} c^{\dagger}_{k\sigma_3} c_{l\sigma_4}. \tag{3.7}$$

The contents of files are shown as below.

```
======================
NInterAll      96
======================
========zInterAll=====
======================
    0     0     0     0     1     0     1     0   0.500000  0.000000
    0     0     0     0     1     1     1     1  -0.500000  0.000000
    0     1     0     1     1     0     1     0  -0.500000  0.000000
    0     1     0     1     1     1     1     1   0.500000  0.000000
    0     0     0     1     1     1     1     0   1.000000  0.000000
    0     1     0     0     1     0     1     1   1.000000  0.000000
...
```

Here, we explain an interaction between a site i and a site j in the case of $S = 1/2$, for simplicity. Using fermion operators, interaction terms for spin operators can be rewritten as

$$\begin{aligned} H_{i,i+1} &= J(S_x^{(i)} S_x^{(i+1)} + S_y^{(i)} S_y^{(i+1)} + S_z^{(i)} S_z^{(i+1)}) \\ &= J\left(\frac{1}{2} S_+^{(i)} S_-^{(i+1)} + \frac{1}{2} S_-^{(i)} S_+^{(i+1)} + S_z^{(i)} S_z^{(i+1)}\right) \\ &= J\left[\frac{1}{2} c^{\dagger}_{i\uparrow} c_{i\downarrow} c^{\dagger}_{i+1\downarrow} c_{i+1\uparrow} + \frac{1}{2} c^{\dagger}_{i\downarrow} c_{i\uparrow} c^{\dagger}_{i+1\uparrow} c_{i+1\downarrow} + \frac{1}{4}(c^{\dagger}_{i\uparrow} c_{i\uparrow} - c^{\dagger}_{i\downarrow} c_{i\downarrow})(c^{\dagger}_{i+1\uparrow} c_{i+1\uparrow} - c^{\dagger}_{i+1\downarrow} c_{i+1\downarrow})\right]. \end{aligned}$$

Thus, the interaction $S_z^{(i)} S_z^{(i+1)}$ for $J = 2$ can be written as

```
    i     0     i     0    i+1     0    i+1     0   0.500000  0.000000
    i     0     i     0    i+1     1    i+1     1  -0.500000  0.000000
    i     1     i     1    i+1     0    i+1     0  -0.500000  0.000000
    i     1     i     1    i+1     1    i+1     1   0.500000  0.000000
```

in the format of an InterAll file. The other terms can be written as below.

```
    i     0     i     1    i+1     1    i+1     0   1.000000  0.000000
    i     1     i     0    i+1     0    i+1     1   1.000000  0.000000
```

There are other file formats for constructing Hamiltonian. The details for input formats of two-body interactions are shown in 4.2.6-4.2.13.

3.2.4 Setting output components

In OneBodyG and TwoBodyG files, calculating components for one-body and two-body Green's functions are defined, respectively.

Setting calculating components for one-body Green's functions

In a OneBodyG file (in this sample, greenone.def), calculating components for $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$ are defined. The contents of files are shown as below.

```
===============================
NCisAjs         32
===============================
======== Green functions ======
===============================
    0     0     0     0
    0     1     0     1
    1     0     1     0
    1     1     1     1
    2     0     2     0
...
```

The details for input formats of a OneBodyG file are shown in 4.2.14.

Setting calculating components for two-body Green's functions

In a TwoBodyG file (in this sample, greentwo.def), calculating components for $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4} \rangle$ are defined. The contents of files are shown as below.

```
=============================================
NCisAjsCktAltDC       1024
=============================================
======== Green functions for Sq AND Nq ======
=============================================
    0     0     0     0     0     0     0     0
    0     0     0     0     0     1     0     1
    0     0     0     0     1     0     1     0
    0     0     0     0     1     1     1     1
    0     0     0     0     2     0     2     0
    ...
```

The details for input formats of a TwoBodyG file are shown in 4.2.15.

3.2.5 Running

After making all input files above, we ready to run a program. For expert mode, we must set an option "-e" and a file name list (in this sample, namelist.def) as arguments to run $\mathcal{H}\Phi$.

$ *Path*/HPhi -e namelist.def

The process after calculating is same as that of standard mode.

3.2.6 Other tutorials

There are following tutorials in `samples/Expert/`.

- The Hubbard model on the two dimensional square lattice (`samples/Expert/Hubbard/square/`)
- The Hubbard model on the two dimensional triangular lattice (`samples/Expert/Hubbard/triangular/`)
- The one dimensional Kondo chain (`samples/Expert/Kondo/chain/`)
- The one dimensional antiferromagnetic Heisenberg chain (`samples/Expert/Spin/HeisenbergChain/HeisenbergChain/`)
- The antiferromagnetic Heisenberg model on the two dimensional square lattice (`samples/Expert/Spin/HeisenbergSquare/`)
- The Kitaev model with 2×3 unit cells to Honeycomb lattice (`samples/Expert/Spin/Kitaev/`)

We can perform these tutorials in the same way to the previous one.

3.3 Making input files for *Expert* mode

This mode is to make input files for *Expert* mode. A set of input files made by this mode gives a model provided in *Standard* mode. The usage is shown as below.

1. Make an input file for *Standard* mode.
2. Setting an option "-sdry" and an input file (in this example, StdFace.def), run $\mathcal{H}\Phi$.
   `$` *Path*`/HPhi -sdry StdFace.def`

   In this case, you should not use MPI parallelization (`mpirun`, `mpiexec`, etc.).
3. Following files are made as input files for *Expert* mode in the present working directory.

```
calcmod.def   greentwo.def  namelist.def  zTrans.def
greenone.def  modpara.def   zInterAll.def zlocspn.def
```

4 File specification

4.1 Input files for *Standard* mode

An example of input file for the standard mode is shown below:

```
W = 2
 L = 4
 model = "spin"
 method = "Lanczos"

 lattice = "triangular lattice"
//mu = 1.0
// t = -1.0
// t’ = -0.5
// U = 8.0
//V = 4.0
//V’=2.0
J = -1.0
J’=-0.5
// nelec = 8
2Sz = 0
```

Basic rules for input files

- In each line, there is a set of a keyword (before an "=") and a parameter(after an "="); they are separated by "=".
- You can describe keywords in a random order.
- Empty lines and lines beginning in a "//"(comment outs) are skipped.
- Upper- and lowercase are not distinguished. Double quotes and blanks are ignored.
- There are three kinds of parameters.
  1. Parameters that must be specified (if not, $\mathcal{H}\Phi$ will stop with error messages),
  2. Parameters that is not necessary be specified (if not, default values are used),
  3. Parameters that must not be specified (if specified, $\mathcal{H}\Phi$ will stop with error messages).

An example of 3 is transfer t for the Heisenberg spin system. If you choose "model=spin", you should not specify "t".

We explain each keywords as follows:

4.1.1 Parameters about the kind of a calculation

- `model`

  Type : String (Choose from `"Fermion Hubbard"`, `"Spin"`, `"Kondo Lattice"`, `"Fermion HubbardGC"`, `"SpinGC"`, `"Kondo LatticeGC"`, `"SpinGCBoost"`) [*1]

  Description : The target model is specified with this parameter; above words denote the canonical ensemble of the Fermion in the Hubbard model

$$H = -\mu \sum_{i\sigma} c_{i\sigma}^\dagger c_{i\sigma} - \sum_{i\neq j\sigma} t_{ij} c_{i\sigma}^\dagger c_{j\sigma} + \sum_i U n_{i\uparrow} n_{i\downarrow} + \sum_{i\neq j} V_{ij} n_i n_j, \tag{4.1}$$

  canonical ensemble in the Spin model($\{\sigma_1, \sigma_2\} = x, y, z$)

$$\begin{aligned} H = &-h \sum_i S_{iz} - \Gamma \sum_i S_{ix} + D \sum_i S_{iz} S_{iz} \\ &+ \sum_{ij,\sigma_1} J_{ij\sigma_1} S_{i\sigma_1} S_{j\sigma_1} + \sum_{ij,\sigma_1\neq\sigma_2} J_{ij\sigma_1\sigma_2} S_{i\sigma_1} S_{j\sigma_2}, \end{aligned} \tag{4.2}$$

  canonical ensemble in the Kondo lattice model

$$H = -\mu \sum_{i\sigma} c_{i\sigma}^\dagger c_{i\sigma} - t \sum_{\langle ij\rangle\sigma} c_{i\sigma}^\dagger c_{j\sigma} + \frac{J}{2} \sum_i \left\{ S_i^+ c_{i\downarrow}^\dagger c_{i\uparrow} + S_i^- c_{i\uparrow}^\dagger c_{i\downarrow} + S_{iz}(n_{i\uparrow} - n_{i\downarrow}) \right\}, \tag{4.3}$$

  grand canonical ensemble of the Fermion in the Hubbard model [Eqn. (4.1)], grand canonical ensemble in the Spin model [Eqn. (4.2)], and grand canonical ensemble in Kondo lattice model [Eqn. (4.3)] respectively.

  When `model="SpinGCBoost"`, by using more efficient algorithm[*2], $\mathcal{H}\Phi$ calculates a system the same as `"SpinGC"`. However, supported models and MPI processes are highly limited. See `"Lattice"` section.

- `method`

  Type : String (Choose from `"Lanczos"`, `"TPQ"`, `"Full Diag"`)

  Description : The calculation type is specified with this parameter; above words denote the single eigenstate calculation by using the Lanczos method, at the finite-temperature by thermally pure quantum state, and the full diagonalization method, respectively.

[*1]GC=Grand Canonical

[*2]Y. Yamaji *et. al.*, manuscript in preparation.

- `lattice`

  Type : String (Choose from `"Chain Lattice"`, `"Square Lattice"`, `"Triangular Lattice"`, `"Honeycomb Lattice"`, `"Ladder"`, `"Kagome"`)

  Description : The lattice shape is specified with this parameter; above words denote the one dimensional chain lattice (Fig. 4.1(a)), the two dimensional square lattice (Fig. 4.1(b)), the two dimensional triangular lattice (Fig. 4.1(c)), the two dimensional anisotropic honeycomb lattice (Fig. 4.2), the ladder lattice (Fig. 4.4), and the Kagome Lattice(Fig. 4.3) respectively.

  In `method="SpinGCBoost"`, only `"Chain Lattice"`, `"Honeycomb Lattice"`, `"Ladder"`, and `"Kagome"` are supported. Limits of L, W, and the number of MPI processes ($N_{\rm proc}$) are as follows:

  - `"Chain Lattice"`
    $L = 8n$ (where n is an integer number under the condition $n \geq 1$), $N_{\rm proc} \leq 2 (L = 8)$, $N_{\rm proc} \leq 2^{L/2-2} (L > 8)$.
  - `"Honeycomb Lattice"`
    $W = 3, L \geq 2$, $N_{\rm proc} \leq 2 (L = 2)$, $N_{\rm proc} \leq 64 (L > 2)$.
  - `"Ladder"`
    $W = 2, L = 2n$ (where n is an integer number under the condition $n \geq 4$), $N_{\rm proc} \leq 2^{L-4}$.
  - `"Kagome"`
    $W = 3, L \geq 2$, $N_{\rm proc} \leq 1 (L = 2)$, $N_{\rm proc} \leq 512 (L > 2)$.

4.1.2 Parameters for the lattice

Chain [Fig. 4.1(a)]

- `L`

  Type : Integer

  Description : The length of the chain is specified with this parameter.

Ladder (Fig. 4.4)

- `L`

  Type : Integer

  Description : The length of the ladder is specified with this parameter.

- `W`

  Type : Integer

  Description : The number of the ladder is specified with this parameter.

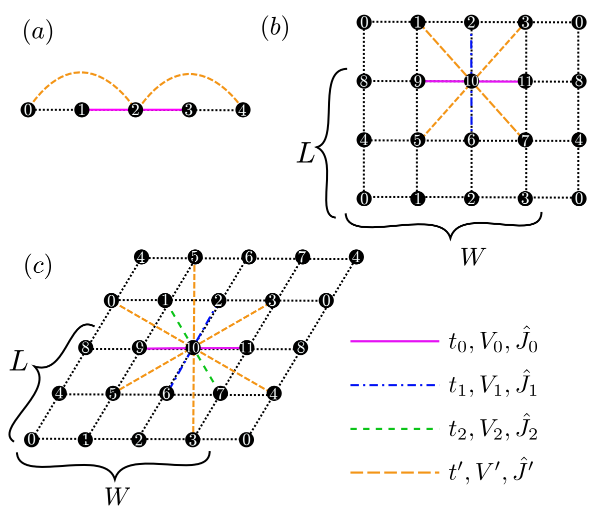

Figure 4.1: Schematic illustration of (a) one dimensional chain lattice, (b) two dimensional square lattice, and (c) two dimensional triangular lattice. They have t, V, and J as a nearest neighbor hopping, an offsite Coulomb integral, and a spin-coupling constant, respectively (magenta solid lines); They also have t', V', and J' as a next nearest neighbor hopping, offsite Coulomb integral, and spin-coupling constant, respectively (green dashed line).

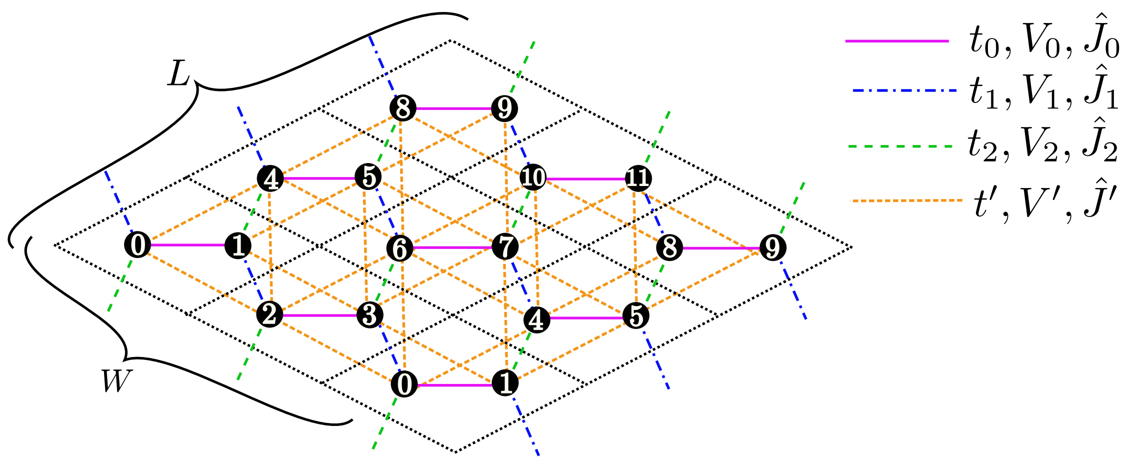

Figure 4.2: Schematic illustration of the anisotropic honeycomb lattice. The nearest neighbor hopping integral, spin coupling, offsite Coulomb integral depend on the bond direction. Those between second nearest neighbor sites are not supported.

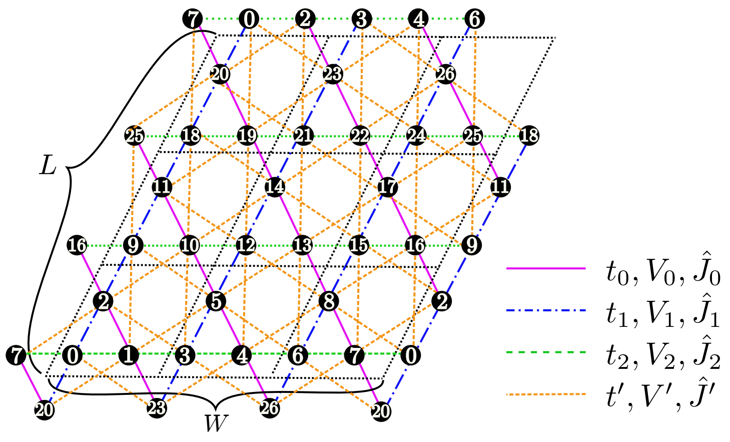

Figure 4.3: Schematic illustration of the Kagome lattice.

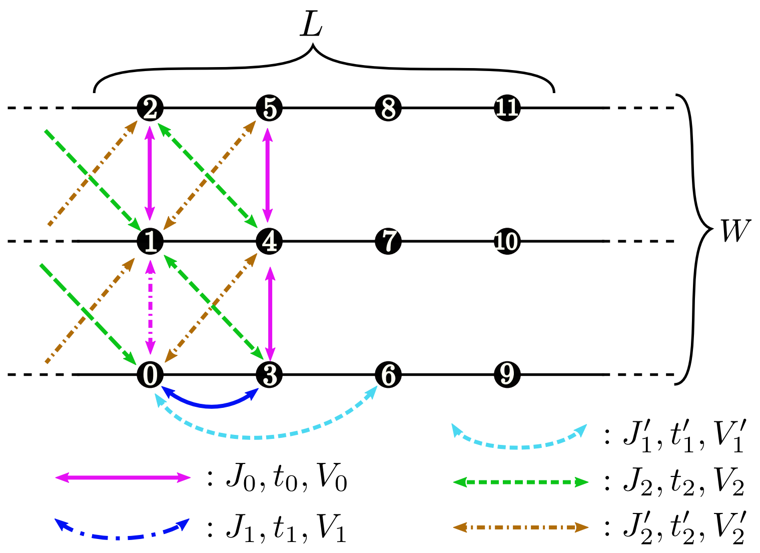

Figure 4.4: Schematic illustration of the ladder lattice.

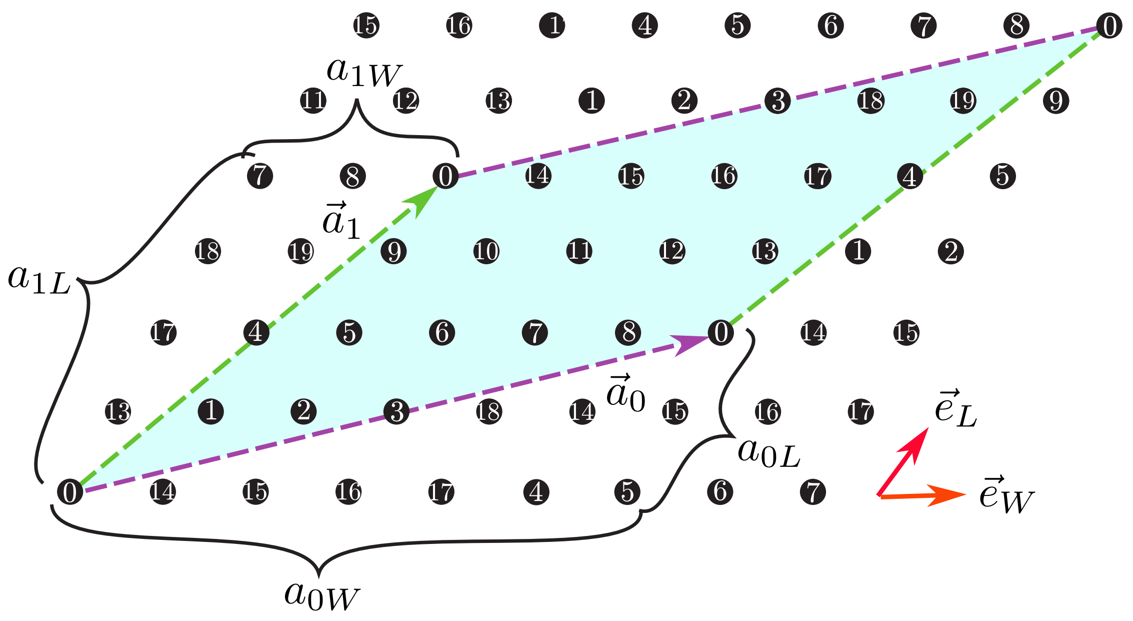

Figure 4.5: The shape of the numerical cell when $\boldsymbol{a}_0 = (6, 2), \boldsymbol{a}_1 = (2, 4)$ in the triangular lattice. The region surrounded by $\boldsymbol{a}_0$(Magenta dashed arrow) and $\boldsymbol{a}_1$(Green dashed arrow) becomes the cell to be calculated (20 sites).

Tetragonal lattice [Fig. 4.1(b)], Triangular lattice[Fig. 4.1(c)], Honeycomb lattice(Fig. 4.2), Kagome lattice(Fig. 4.3)

In these lattices, we can specify the shape of the numerical cell by using the following two methods.

- `W`, `L`

  Type : Integer

  Description : The alignment of original unit cells (dashed black lines in Figs. 4.1 - 4.3) is specified with these parameter.

- `a0W`, `a0L`, `a1W`, `a1L`

  Type : Integer

  Description : We can specify two vectors $(\boldsymbol{a}_0, \boldsymbol{a}_1)$ that surrounds the numerical cell (Fig. 4.5). These vectors should be specified in the Fractional coordinate.

If we use both of these method, $\mathcal{H}\Phi$ stops. When `model=SpinGCBoost`, we can use only the former.

We can check the shape of the numerical cell by using a file `lattice.gp` which is written in the Standard mode. This file can be read by `gnuplot` as follows:

```
$ gnuplot lattice.gp
```

4.1.3 Parameters for conserved quantities

- `nelec`

  Type : Positive integer

  Description : The number of valence electrons is specified with this parameter. When `model = "Fermion HubbardGC"`, `"Spin"`, or `"SpinGC"`, it should not be specified.

- `2Sz`

  Type : Integer

  Description : The z component of the twofold total spin is specified with this parameter. When `model = "Fermion HubbardGC"` or `SpinGC`, it should not be specified.

4.1.4 Parameters for the Hamiltonian

A default value is set as 0 unless a specific value is not defined in a description. Table 4.1 shows the parameters for each models. In the case of a complex type, a file format is "*a real part, an imaginary part*" while in the case of a real type, only "*a real part*".

Local terms

- `mu`

  Type : Real

  Description : (Hubbard and Kondo lattice model) The chemical potential μ (including the site potential) is specified with this parameter.

- `U`

  Type : Real

  Description : (Hubbard and Kondo lattice model) The onsite Coulomb integral U is specified with this parameter.

- `Jx`, `Jy`, `Jz`, `Jxy`, `Jyx`, `Jxz`, `Jzx`, `Jyz`, `Jzy`

  Type : Real

  Description : (Kondo lattice model) The spin-coupling constant between the valence and the local electrons is specified with this parameter. If the exchange coupling `J` is specified in the input file, instead of `Jx, Jy, Jz`, the diagonal exchange couplings, `Jx, Jy, Jz`, are set as `Jx = Jy = Jz = J`. When both the set of exchange couplings (`Jx`, `Jy`, `Jz`) and the exchange coupling `J` are specified in the input file, $\mathcal{H}\Phi$ will stop.

- `h`, `Gamma`, `D`

  Type : Real

  Description : (Spin model) The longitudinal magnetic field, transverse magnetic field, and the single-site anisotropy parameter are specified with these parameters. The single-site anisotropy parameter is not available for `model=SpinGCBoost`.

The non-local terms described below should be specified in different ways depending on the lattice structure: For `lattice=Ladder`, the non-local terms are specified in the different way from `lattice=Chain Lattice`, `Square Lattice`, `Triangular Lattice`, `Honeycomb Lattice`, `Kagome`. Below, the available parameters for each lattice are shown in Table 4.1.

Interactions	1D chain	2D square	2D triangular	Honeycomb	Kagome	Ladder
`J`, `t`, `V` (simplified)	○	○	○	○	○	-
`J'`, `t'`, `V'`	○	○	○	○	○	-
`J0`, `t0`, `V0`	○	○	○	○	○	○
`J1`, `t1`, `V1`	-	○	○	○	○	○
`J2`, `t2`, `V2`	-	-	○	○	○	○
`J1'`, `t1'`, `V1'`	-	-	-	-	-	○
`J2' ,t2'`, `V2'`	-	-	-	-	-	○

Table 4.1: Interactions for each models defined in an input file. We can define spin couplings as matrix format.

Non-local terms[for Ladder (Fig. 4.4)]

- `t0`, `t1`, `t1'`, `t2`, `t2'`

  Type : Complex

  Description : (Hubbard and Kondo lattice model) Hopping integrals in the ladder lattice (See Fig. 4.4) is specified with this parameter.

- `V0`, `V1`, `V1'`, `V2`, `V2'`

  Type : Real

  Description : (Hubbard and Kondo lattice model) Offsite Coulomb integrals on the ladder lattice (Fig. 4.2 are specified with these parameters.

- `J0x`, `J0y`, `J0z`, `J0xy`, `J0yx`, `J0xz`, `J0zx`, `J0yz`, `J0zy`

- `J1x`, `J1y`, `J1z`, `J1xy`, `J1yx`, `J1xz`, `J1zx`, `J1yz`, `J1zy`

- `J1'x`, `J1'y`, `J1'z`, `J1'xy`, `J1'yx`, `J1'xz`, `J1'zx`, `J1'yz`, `J1'zy`

- `J2x`, `J2y`, `J2z`, `J2xy`, `J2yx`, `J2xz`, `J2zx`, `J2yz`, `J2zy`

- `J2'x`, `J2'y`, `J2'z`, `J2'xy`, `J2'yx`, `J2'xz`, `J2'zx`, `J2'yz`, `J2'zy`

  Type : Real

  Description : (Spin model) Spin-coupling constants in the ladder lattice (See Fig. 4.4) are specified with these parameter. If the simplified parameter `J0` is specified in the input file instead of the diagonal couplings, `J0x, J0y, J0z`, these diagonal couplings are set as `J0x = J0y = J0z = J0`. If both `J0` and the set of the couplings (`J0x, J0y, J0z`) are specified, $\mathcal{H}\Phi$ will stop. The above rules are also valid for the simplified parameters, `J1`, `J1'`, `J2`, and `J2'`.

Non-local terms [other than Ladder (Figs. 4.1, 4.2, 4.3)]

- `t0`, `t1`, `t2`

  Type : Complex

  Description : (Hubbard and Kondo lattice model) Nearest neighbor hoppings for each direction (See Figs. 4.1-4.3) are specified with these parameter. If there is no bond dependence of the hoppings, the simplified parameter `t` is available to specify `t0`, `t1`, and `t2` as `t0 = t1 = t2 = t`. If both `t` and the set of the hoppings (`t0`, `t1`, `t2`) are specified, $\mathcal{H}\Phi$ will stop.

- `V0`, `V1`, `V2`

  Type : Real

  Description : (Hubbard and Kondo lattice model) Nearest-neighbor offsite Coulomb integrals V for each direction (See Figs. 4.1-4.3) are specified with these parameters. If there is no bond dependence of the offsite Coulomb integrals, the simplified parameter `V` is available to specify `V0`, `V1`, and `V2` as `V0 = V1 = V2 = V`. If both `V` and the set of the Coulomb integrals (`V0`, `V1`, `V2`) are specified, $\mathcal{H}\Phi$ will stop.

- `J0x`, `J0y`, `J0z`, `J0xy`, `J0yx`, `J0xz`, `J0zx`, `J0yz`, `J0zy`

- `J1x`, `J1y`, `J1z`, `J1xy`, `J1yx`, `J1xz`, `J1zx`, `J1yz`, `J1zy`

- `J2x`, `J2y`, `J2z`, `J2xy`, `J2yx`, `J2xz`, `J2zx`, `J2yz`, `J2zy`

  Type : Real

  Description : (Spin model) Nearest-neighbor exchange couplings for each direction are specified with thees parameters. If the simplified parameter `J0` is specified, instead of `J0x, J0y, J0z`, the exchange couplings, `J0x, J0y, J0z`, are set as `J0x = J0y = J0z = J0`. If both `J0` and the set of the exchange couplings (`J0x, J0y, J0z`) are specified, $\mathcal{H}\Phi$ will stop. The above rules are valid for `J1` and `J2`.

  If there is no bond dependence of the exchange couplings, the simplified parameters, `Jx`, `Jy`, `Jz`, `Jxy`, `Jyx`, `Jxz`, `Jzx`, `Jyz`, `Jzy`, are available to specify the exchange couplings for every bond as `J0x = J1x = J2x = Jx`. If any simplified parameter (`Jx`$\sim$`Jzy`) is specified in addition to its counter parts

(`J0x`∼`J2zy`), $\mathcal{H}\Phi$ will stop. Below, examples of parameter sets for nearest-neighbor exchange couplings are shown.

- If there are no bond-dependent, no anisotropic and offdiagonal exchange couplings (such as J_{xy}), please specify `J` in the input file.
- If there are no bond-dependent and offdiagonal exchange couplings but are anisotropic couplings, please specify the non-zero couplings in the diagonal parameters, `Jx, Jy, Jz`.
- If there are no bond-dependent exchange couplings but are anisotropic and offdiagonal exchange couplings, please specify the non-zero couplings in the nine parameters, `Jx, Jy, Jz, Jxy, Jyz, Jxz, Jyx, Jzy, Jzx`.
- If there are no anisotropic and offdiagonal exchange couplings, but are bond-dependent couplings, please specify the non-zero couplings in the three parameters, `J0, J1, J2`.
- If there are no anisotropic exchange couplings, but are bond-dependent and offdiagonal couplings, please specify the non-zero couplings in the nine parameters, `J0x, J0y, J0z, J1x, J1y, J1z, J2x, J2y, J2z`.
- If there are bond-dependent, anisotropic and offdiagonal exchange couplings, please specify the non-zero couplings in the twenty-seven parameters from `J0x` to `J2zy`.

- `t'`

  Type : Complex

  Description : (Hubbard and Kondo lattice model) Nearest neighbor hoppings for each direction (See Figs. 4.1-4.3) are specified with these parameter.

- `V'`

  Type : Real

  Description : (Hubbard and Kondo lattice model) Nearest neighbor-offsite Coulomb integrals V for each direction (See Figs. 4.1-4.3) are specified with these parameters.

- `J'x`, `J'y`, `J'z`, `J'xy`, `J'yx`, `J'xz`, `J'zx`, `J'yz`, `J'zy`

  Type : Real

  Description : (Spin model) Second nearest-neighbor exchange couplings are specified. However, for `lattice = Honeycomb Lattice` and `lattice = Kagome` with `model=SpinGCBoost`, the second nearest-neighbor exchange couplings are not available in the *Standard* mode. If the simplified parameter `J'` is specified, instead of `J'x, J'y, J'z`, the exchange couplings are set as `J'x = J'y = J'z = J'`. If both `J'` and the set of the couplings (`J'x, J'y, J'z`), $\mathcal{H}\Phi$ will stop.

4.1.5 Parameters for the numerical condition

- `Lanczos_max`

  Type : Positive integer (`2000` as a default)

  Description : Upper limit of the Lanczos step is specified with this parameter.

- `initial_iv`

  Type : Integer (`-1` as a default)

  Description : An initial vector is specified with this parameter.

  - Lanczos method
    * For canonical ensemble and `initial_iv` ≥ 0
      The non-zero components of an initial vector is specified with this parameter.
    * For grand canonical ensemble or `initial_iv` < 0
      A seed of random generator is given by this parameter and the random vector is used as an initial vector.
  - TPQ method
    A seed of random generator is given by this parameter and the random vector is used as an initial vector.

  See 5 for details of setting an initial vector.

- `exct`

  Type : Positive integer (`1` as a default)

  Description : We specify the number of getting eigenvectors from the ground energy by Lanczos method.
  When exct=2, we obtain the eigenvector of the first-excited state.

  Note: the following condition must be satisfied: `nvec` >= `exct`.

- `LanczosEps`

  Type : Positive Integer (`14` as a default)

  Description : The convergence criteria for the Lanczos method is specified with this parameter. If the difference between the old and the new target eigenvalue fall below $10^{-\texttt{LanczosEps}}$, the Lanczos step will finish.

- `LancczosTarget`

  Type : Positive integer (`2` as a default)

  Description : We specify the target eigenenergy for the convergence criteria. If this set to `1`, target eigenenergy becomes the ground state.

- `LargeValue`

  Type : Double (The default value is written below.)

  Description : (Only for TPQ) We use l as $l - \hat{H}/N_s$ in the TPQ calculation. Usually, the largest eigenvalue of Hamiltonian is used as l. Thus, we take the default value of l as a summation of the absolute values of each coefficients in the Hamiltonian divided by the number of sites.

- `NumAve`

  Type : Positive integer (`5` as a default)

  Description : (Only for the TPQ) The number of independent runs for the TPQ method is specified with this parameter.

- `ExpecInterval`

  Type : Positive integer (`20` as a default)

  Description : (Only for the TPQ) We specify the interval steps of calculating correlation functions in TPQ method.
  Note: The small interval increases the time cost of calculations.

- `OutputMode`

  Type : Choose from `"none"`, `"correlation"`, and `"full"` (`correlation` as a default)

  Description : Indices of correlation functions are specified with this keyword. `"none"` indicates correlation functions will not calculated. When `outputmode="correlation"`, $\langle c^\dagger_{i\sigma} c_{i\sigma}\rangle$ is computed at all i, σ, and $\langle c^\dagger_{i\sigma} c_{i\sigma} c^\dagger_{j\sigma'} c_{j\sigma'}\rangle$ is computed at all i, j, σ, σ'. If `"full"` is selected, $\langle c^\dagger_{i\sigma} c_{j\sigma'}\rangle$ is computed at all i, j, σ, σ', and $\langle c^\dagger_{i_1\sigma_1} c_{i_2\sigma_2} c^\dagger_{i_3\sigma_3} c_{i_4\sigma_4}\rangle$ is computed at all $i_1, i_2, i_3, i_4, \sigma_1, \sigma_2, \sigma_3, \sigma_4$.

  In spin system, indices are specified as those on the Bogoliubov representation (See 5.4).

4.2 Input files for *Expert* mode

In this section, details of input files for expert mode are explained. Input files are categorized by the following four parts.

(1) List: This file is a list of input file names with keywords. Each keywords is fixed, but file names are free to be determined.

(2) Basic parameters: The following input files give basic parameters. The kinds of input files are determined by keywords.
CalcMod: Set the parameters for calculation modes.
ModPara: Set the parameters for basic parameters such as site number, electron number, Lanczos step *etc.*
LocSpin: Set the location of local spin (only used in Kondo model).

(3) Hamiltonian: Hamiltonian for $\mathcal{H}\Phi$ is denoted by the format of interactions for electron system. The kinds of interactions are determined by the following keywords.
Trans: The one body part, $c^\dagger_{i\sigma_1}c_{j\sigma_2}$.
InterAll: The general two body interactions, $c^\dagger_{i\sigma_1}c_{j\sigma_2}c^\dagger_{k\sigma_3}c_{l\sigma_4}$.
We can set interactions which are often used by the following keywords.
CoulombIntra: On-site Coulomb interactions, $n_{i\uparrow}n_{i\downarrow}$ ($n_{i\sigma} = c^\dagger_{i\sigma}c_{i\sigma}$).
CoulombInter: Off-site Coulomb interactions, n_in_j ($n_i = n_{i\uparrow} + n_{i\downarrow}$).
Hund: Hund couplings, $n_{i\uparrow}n_{j\uparrow} + n_{i\downarrow}n_{j\downarrow}$.
PairHop: Pair hopping couplings, $c^\dagger_{i\uparrow}c_{j\uparrow}c^\dagger_{i\downarrow}c_{j\downarrow}$.
Exchange: Exchange couplings, $c^\dagger_{i\uparrow}c_{j\uparrow}c^\dagger_{j\downarrow}c_{i\downarrow}$.
Ising: Ising interactions, $S^z_iS^z_j$.
PairLift: PairLift couplings, $c^\dagger_{i\uparrow}c_{i\downarrow}c^\dagger_{j\uparrow}c_{j\downarrow}$.

(4) Output: Targets for output is determined.
OneBodyG : One-body green functions, $\langle c^\dagger_{i\sigma_1}c_{j\sigma_2}\rangle$.
TwoBodyG : Two-body green functions, $\langle c^\dagger_{i\sigma_1}c_{j\sigma_2}c^\dagger_{k\sigma_3}c_{l\sigma_4}\rangle$.

4.2.1 List file for Input files

. This file determines input filenames which are correlated with keywords. File format is shown as follows.

```
CalcMod  calcmod.def
ModPara  modpara.def
LocSpin  zlocspn.def
Trans    ztransfer.def
InterAll zinterall.def
OneBodyG zcisajs.def
TwoBodyG zcisajscktaltdc.def
```

File format

[string01] [string02]

Parameters

- [string01]

  Type : string-type

  Description : Select a word from keywords.

- [string02]

  Type : string-type

Description : An input filename which is correlated with keywords.

Use rules

- After setting keywords at [string 01], half-width state is needed for writing a filename. You can set the filename freely.
- Keywords for input files are shown in Table4.2.
- Essential keywords are "CalcMod", "ModPara" and "LocSpin".
- Keywords can be set in random order.
- If keywords or filenames are incorrect, the program is terminated.
- When the head of line is "#", the line is skipped.

Keywords	Details for corresponding files
CalcMod	Parameters for modes of calculation.
ModPara	Parameters for calculation.
LocSpin	Configurations of the local spins for Hamiltonian.
Trans	Transfer and chemical potential for Hamiltonian.
InterAll	Two-body interactions for Hamiltonian.
CoulombIntra	CoulombIntra interactions.
CoulombInter	CoulombInter interactions.
Hund	Hund couplings.
PairHop	Pair hopping couplings.
Exchange	Exchange couplings.
Ising	Ising interactions.
PairLift	Pair lift couplings.
OneBodyG	Output components for Green functions $\langle c_{i\sigma}^\dagger c_{j\sigma}\rangle$
TwoBodyG	Output components for Correlation functions $\langle c_{i\sigma}^\dagger c_{j\sigma} c_{k\tau}^\dagger c_{l\tau}\rangle$
SingleExcitation	Operators for generating single excited state
PairExcitation	Operators for generating pair excited state
SpectrumVec	An input vector to calculate a restart vector

Table 4.2: List of the definition files.

4.2.2 CalcMod file

This file determines parameters for calculation method, model and output mode. File format is shown as follows.

```
CalcType   0
CalcModel   2
CalcEigenVec 0
```

File format

[string01] [int01]

Parameters

- [string01]

  Type : string-type

  Description : Select a word from keywords.

- [int01]

  Type : int-type

  Description : A parameter which is correlated with a keyword.

Use rules

- After setting keywords at [string 01], a half-width blank is needed for setting a parameter.
- Keywords can be set in random order.
- If keywords or filenames are incorrect, the program is terminated.
- The following keywords, “CalcType” and “CalcModel” are essential.
- When the head of line is “#”, the line is skipped.

Keywords and parameters

Parameters correlated with keywords are shown as follows.

- `CalcType`

  Type : int-type

  Description : Select the method for calculation from the following list:
  0: Lanczos method,

1: Analysis of the physical properties by using TPQ,
2: Full diagonalization method.
3. Calculate dynamical Green's functions.

- `CalcModel`

  Type : int-type

  Description : Select the model from the following list:
  0: fermion Hubbard model (Canonical ensemble: conservation of particles or conservation of particles and the component of S_z)
  1: Spin model (Canonical ensemble: conservation of the component of S_z)
  2: Kondo lattice model (Canonical ensemble: conservation of particles, the component of S_z)
  3: fermion Hubbard model (Grand canonical ensemble)
  4: Spin model (Grand canonical ensemble)
  5: Kondo lattice model (Grand canonical ensemble) For fermion Hubbard model, you can select the model under the conservation of particles by setting `NCond` in the ModPara file. When you want to select the model under the conservation of particles and the component of S_z, you set both `NCond` and `2Sz` in the ModPara file.

- `CalcEigenVec`

  Type : int-type (default value: 0)

  Description : Select the method to calculate eigenvectors:
  0:Lanczos+CG methods (When the convergence of eigenvectors are not enough for using Lanczos method, CG method is applied to calculate eigenvectors).
  1:Lanczos method.

- `InitialVecType`

  Type : int-type (default value: 0)

  Description : Select the type of an initial vector:
  0:Complex type.
  1:Real type.

- `OutputEigenVec`

  Type : int-type (default value: 0)

  Description : Select the mode of outputting an eigenvector:
  0: not output an eigen vector.
  1: output an eigen vector.

- `InputEigenVec`

  Type : int-type (default value: 0)

Description : Select the mode of inputting an eigenvector:
0: not input an eigen vector.
1: input an eigen vector.

- `ReStart`

  Type : int-type (default value: 0)

  Description : Select the mode of inputting a restart vector:
  0: not restart calculation,
  1: output a restart vector,
  2: input a restart vector and output a new restart vector,
  3: input a restart vector.

- `CalcSpec`

  Type : int-type (default value: 0)

  Description : Select the mode of calculating dynamical Green's functions.
  0: not calculate dynamical Green's functions,
  1: (not restart) input an initial vector and files for generating single excited or pair excited states,
  2: input components of triangular diagonal matrix,
  3: output both components of triangular diagonal matrix and a restart vector,
  4: input both components of triangular diagonal matrix and a restart vector,
  5: input and output both components of triangular diagonal matrix and a restart vector.

4.2.3 ModPara file

This file determines parameters for calculation. File format is shown as follows.

```
--------------------
Model_Parameters   0
--------------------
VMC_Cal_Parameters
--------------------
CDataFileHead  zvo
CParaFileHead  zqp
--------------------
Nsite          16
Ncond          16
2Sz            0
Lanczos_max    1000
initial_iv     12
exct           1
LanczosEps     14
LanczosTarget  2
LargeValue     12
NumAve         5
ExpecInterval  20
```

File format

- Lines 1-4: Header
- Line 6: [string01] [string02]
- Lines 7-8: Header
- Lines 9- : [string01] [int01]

Parameters

- [string01]

  Type : string-type

  Description : Select a word from keywords.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : Set a header for output files.

- [int01]

  Type : int-type (blank parameter not allowed)

Description : A parameter which is correlated with a keyword.

Use rules

- From Line 9: After setting keywords at [string 01], a half-width blank is needed for setting a parameter.
- All Parameters are needed and the order for parameters is fixed.

Keywords and parameters

- `CDataFileHead`

  Type : string-type (blank parameter not allowed)

  Description : A header for output files. For example, the output filename for one body Green's function becomes "**xxx_Lanczos_cisajs.dat**" (xxx are characters set by `CDataFileHead`).

- `Nsite`

  Type : int-type (Positive integer)

  Description : The number of sites.

- `Ncond`

  Type : int-type (Positive integer)

  Description : The number of conduction electrons (not used in grand canonical ensemble).

- `2Sz`

  Type : int-type (Positive integer)

  Description : The total value of $2S_z$ (not used in grand canonical ensemble). For conservation of S_z in the case of `CalcModel` =0 (fermion Hubbard model) or 2 (Kondo lattice model), we must set `Ncond`.

- `Lanczos_max`

  Type : int-type (Positive integer)

  Description : The number of Lanczos steps in calculation. When the convergence within the specified accuracy is satisfied, the calculation is finished before a step becomes `Lanczos_max`.

- `initial_iv`

  Type : int-type

  Description : An initial vector is specified with this parameter.

- Lanczos method
  - For canonical ensemble and `initial_iv` ≥ 0
    The non-zero components of an initial vector is specified with this parameter.
  - For grand canonical ensemble or `initial_iv` < 0
    A seed of random generator is given by this parameter and the random vector is used as an initial vector.
- TPQ method
  A seed of random generator is given by this parameter and the random vector is used as an initial vector.

See 5 for details of setting an initial vector.

- `exct`

  Type : int-type (Positive integer)

  Description : An integer for setting the number of getting eigenvectors from the ground energy by Lanczos method.

- `LanczosEps`

  Type : int-type (Positive integer)

  Description : An integer for judging a convergence of Lanczos method. The convergence is judged by satisfying the condition that the relative error between an eigenvalue and an eigenvalue at the Lanczos step of the one step before becomes less than $10^{-\texttt{LanczosEps}}$.

- `LanczosTarget`

  Type : int-type (Positive integer)

  Description : An integer giving the target of eigenvalue for judging the convergence of Lanczos method. For example, the target becomes a ground state when `LanczosTarget` is equal to one, and a 1st excited state when `LanczosTarget` is equal to two.

- `LargeValue`

  Type : double-type

  Description : (Only use for TPQ method) An integer giving l of $l - \hat{H}/N_s$ used in TPQ method.

- `NumAve`

  Type : int-type

  Description : (Only use for TPQ method) An integer giving the number of independent runs for TPQ method.

- `ExpecInterval`

  Type : int-type

  Description : (Only use for TPQ method) An integer giving the interval steps of calculating correlation functions in TPQ method.
  Note: The small interval increases the time cost of calculations.

- `OmegaMax`

  Type : double-type

  Description : (Only use for calculating dynamical green's functions) The maximum value of the real part of the frequency $\omega_{\rm max}$ to calculate dynamical green's functions.

- `OmegaMin`

  Type : double-type

  Description : (Only use for calculating dynamical green's functions) The minimum value of the real part of the frequency $\omega_{\rm min}$ to calculate dynamical green's functions.

- `OmegaIm`

  Type : double-type

  Description : (Only use for calculating dynamical green's functions) The imaginary part of the frequency $\eta$to calculate dynamical green's functions.

- `NOmega`

  Type : int-type

  Description : (Only use for calculating dynamical green's functions) The integer to define the step size of the frequency $\Delta\omega = (\omega_{\rm max} - \omega_{\rm min})/N_\omega$. The frequency is given by $z_n = \omega_{\rm min} + \Delta\omega \times n + i\eta$.

4.2.4 LocSpin file

This file determines sites with localized spins. File format is shown as follows.

```
================================
NlocalSpin     6
================================
========i_0LocSpn_1IteElc ======
================================
    0      1
    1      0
    2      1
    3      0
    4      1
    5      0
    6      1
    7      0
    8      1
    9      0
   10      1
   11      0
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of localized spins. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of localized spins.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02] < `Nsite`).

- [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer for selecting an electron state whether localized spin or itinerant electron states:
  0: Itinerant electron state,
  n >0: localized spin state with $2S = n$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of localized spins indicated by [int03].
- A program is terminated, when [int02] is different from the total number of sites.
- A program is terminated under the condition [int02] $<$ 0 or `Nsite` $<=$ [int02].

4.2.5 Trans file

This file determines values of transfer integrals $t_{ij\sigma_1\sigma2}$,

$$H+ = -\sum_{ij\sigma_1\sigma2} t_{ij\sigma_1\sigma2} c_{i\sigma_1}^{\dagger} c_{j\sigma_2}. \tag{4.4}$$

An example of file format is shown as follows.

```
========================
NTransfer      24
========================
========i_j_s_tijs======
========================
    0     0     2     0   1.000000  0.000000
    2     0     0     0   1.000000  0.000000
    0     1     2     1   1.000000  0.000000
    2     1     0     1   1.000000  0.000000
    2     0     4     0   1.000000  0.000000
    4     0     2     0   1.000000  0.000000
    2     1     4     1   1.000000  0.000000
    4     1     2     1   1.000000  0.000000
    4     0     6     0   1.000000  0.000000
    6     0     4     0   1.000000  0.000000
    4     1     6     1   1.000000  0.000000
    6     1     4     1   1.000000  0.000000
    6     0     8     0   1.000000  0.000000
    8     0     6     0   1.000000  0.000000
...
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [int05] [double01] [double02]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of transfer integrals. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of transfer integrals.

- [int02], [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int04] < `Nsite`).

- [int03], [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a spin index,
  0: up-spin,
  1: down-spin.

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for a real part of $t_{ij\sigma_1\sigma_2}$.

- [double02]

  Type : double-type (blank parameter not allowed)

  Description : A value for an imaginary part of $t_{ij\sigma_1\sigma_2}$.

Use rules

- Headers cannot be omitted.
- Since Hamiltonian must be Hermitian, the following relation must be satisfied, $t_{ij\sigma_1\sigma_2} = t^{\dagger}_{ji\sigma_2\sigma_1}$. A program is terminated when the above relation is broken.
- A program is terminated, when components of on-site interactions are double counted.
- A program is terminated, when [int01] is different from the total number of transfer integrals defined in this file.
- A program is terminated, when [int02]-[int05] are out of range from the defined values.

4.2.6 InterAll file

This file determines values of generalized two body interactions integrals $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}$,

$$H+=\sum_{i,j,k,l}\sum_{\sigma_1,\sigma_2,\sigma_3,\sigma_4} I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4}. \tag{4.5}$$

For Spin, the condition $i=j$ and $k=l$ must be satisfied. An example of file format is shown as follows.

```
======================
NInterAll      36
======================
========zInterAll=====
======================
0    0    0    1    1    1    1    0   0.50  0.0
0    1    0    0    1    0    1    1   0.50  0.0
0    0    0    0    1    0    1    0   0.25  0.0
0    0    0    0    1    1    1    1  -0.25  0.0
0    1    0    1    1    0    1    0  -0.25  0.0
0    1    0    1    1    1    1    1   0.25  0.0
2    0    2    1    3    1    3    0   0.50  0.0
2    1    2    0    3    0    3    1   0.50  0.0
2    0    2    0    3    0    3    0   0.25  0.0
2    0    2    0    3    1    3    1  -0.25  0.0
2    1    2    1    3    0    3    0  -0.25  0.0
2    1    2    1    3    1    3    1   0.25  0.0
4    0    4    1    5    1    5    0   0.50  0.0
4    1    4    0    5    0    5    1   0.50  0.0
4    0    4    0    5    0    5    0   0.25  0.0
4    0    4    0    5    1    5    1  -0.25  0.0
4    1    4    1    5    0    5    0  -0.25  0.0
4    1    4    1    5    1    5    1   0.25  0.0
...
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [int05] [int06] [int07] [int08] [int09] [double01] [double02]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of generalized two body interactions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of generalized two body interactions.

- [int02], [int04], [int06], [int08]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int04], [int06], [int08] < `Nsite`).

- [int03], [int05], [int07], [int09]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a spin index,
  0: up-spin,
  1: down-spin.

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for a real part of $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}$.

- [double02]

  Type : double-type (blank parameter not allowed)

  Description : A value for an imaginary part of $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}$.

Use rules

- Headers cannot be omitted.

- Since Hamiltonian must be Hermitian, the following relation must be satisfied, $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} = I^{\dagger}_{lkji\sigma_4\sigma_3\sigma_2\sigma_1}$. A program is terminated when the above relations are broken. It is noted that a term of Hermitian conjugate for $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c^{\dagger}_{i\sigma_1} c_{j\sigma_2} c^{\dagger}_{k\sigma_3} c_{l\sigma_4}$ should be inputted as $I_{lkji\sigma_4\sigma_3\sigma_2\sigma_1}\ c^{\dagger}_{l\sigma_4} c_{k\sigma_3} c^{\dagger}_{j\sigma_2} c_{i\sigma_1}$.

- A program is terminated, when $i = j$ and $k = l$ are not satisfied for spin model.

- A program is terminated, when components of on-site interactions are double counted.

- A program is terminated, when [int01] is different from the total number of generalized two body interactions defined in this file.
- A program is terminated, when [int02]-[int09] are out of range from the defined values.

4.2.7 CoulombIntra file

This file determines values of on-site interactions U_i (for $S = 1/2$ system),

$$H+ = \sum_i U_i n_{i\uparrow} n_{i\downarrow}. \tag{4.6}$$

An example of file format is shown as follows.

```
======================
NCoulombIntra 6
======================
========i_0LocSpn_1IteElc ======
======================
   0  4.000000
   1  4.000000
   2  4.000000
   3  4.000000
   4  4.000000
   5  4.000000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of on-site interactions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of on-site interactions.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02] < `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for U_i.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of on-site interactions are double counted.
- A program is terminated, when [int01] is different from the total number of on-site interactions defined in this file.
- A program is terminated, when [int02] is out of range from the defined values.

4.2.8 CoulombInter file

This file determines values of off-site interactions V_{ij} (for $S = 1/2$ system),

$$H+ = \sum_{i,j} V_{ij} n_i n_j. \tag{4.7}$$

An example of file format is shown as follows.

```
======================
NCoulombInter 6
======================
========CoulombInter ======
======================
   0     1  1.0000
   1     2  1.0000
   2     3  1.0000
   3     4  1.0000
   4     5  1.0000
   5     0  1.0000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of off-site interactions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of off-site interactions.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int03] < `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for V_{ij}.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of off-site interactions are double counted.
- A program is terminated, when [int01] is different from the total number of off-site interactions defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.9 Hund file

This file determines values of Hund couplings $J_{ij}^{\rm Hund}$ (for $S = 1/2$ system),

$$H+ = -\sum_{i,j} J_{ij}^{\rm Hund}(n_{i\uparrow}n_{j\uparrow} + n_{i\downarrow}n_{j\downarrow}). \tag{4.8}$$

An example of file format is shown as follows.

```
======================
NHund 6
======================
========Hund ======
======================
   0     1 -0.250000
   1     2 -0.250000
   2     3 -0.250000
   3     4 -0.250000
   4     5 -0.250000
   5     0 -0.250000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of Hund couplings. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of Hund couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int03] < `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for $J_{ij}^{\rm Hund}$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of Hund couplings are double counted.
- A program is terminated, when [int01] is different from the total number of Hund couplings defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.10 PairHop file

This file determines values of PairHop couplings $J_{ij}^{\rm Pair}$ (for $S = 1/2$ system),

$$H+ = \sum_{i,j} J_{ij}^{\rm Pair} c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow}. \tag{4.9}$$

An example of file format is shown as follows.

```
======================
NPairhop 12
======================
========Pairhop ======
======================
   0     1  0.50000
   1     0  0.50000
   1     2  0.50000
   2     1  0.50000
   2     3  0.50000
   3     2  0.50000
   3     4  0.50000
   4     3  0.50000
   4     5  0.50000
   5     4  0.50000
   5     0  0.50000
   0     5  0.50000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of PairHop couplings. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of PairHop couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index ($0 <=$ [int02], [int03] $<$ `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for $J_{ij}^{\rm Pair}$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of PairHop couplings are double counted.
- A program is terminated, when [int01] is different from the total number of PairHop couplings defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.11 Exchange file

This file determines values of Exchange couplings $J_{ij}^{\rm Ex}$ (for $S = 1/2$ system). For fermion electronic system, exchange terms are given as

$$H+=\sum_{i,j} J_{ij}^{\rm Ex}(c_{i\uparrow}^{\dagger}c_{j\uparrow}c_{j\downarrow}^{\dagger}c_{i\downarrow}+c_{i\downarrow}^{\dagger}c_{j\downarrow}c_{j\uparrow}^{\dagger}c_{i\uparrow}). \tag{4.10}$$

While for Spin system, they are given as

$$H+=\sum_{i,j} J_{ij}^{\rm Ex}(S_i^+S_j^- + S_i^-S_j^+). \tag{4.11}$$

We note that $(S_i^+S_j^- + S_i^-S_j^+)$ in Spin system is written by the operators for electrons as $-(c_{i\uparrow}^{\dagger}c_{j\uparrow}c_{j\downarrow}^{\dagger}c_{i\downarrow}+c_{i\downarrow}^{\dagger}c_{j\downarrow}c_{j\uparrow}^{\dagger}c_{i\uparrow})$. An example of file format is shown as follows.

```
======================
NExchange 6
======================
========Exchange ======
======================
   0      1  0.50000
   1      2  0.50000
   2      3  0.50000
   3      4  0.50000
   4      5  0.50000
   5      0  0.50000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of Exchange couplings. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of Exchange couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int03] < `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for J_{ij}^{Ex}.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of Exchange couplings are double counted.
- A program is terminated, when [int01] is different from the total number of Exchange couplings defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.12 Ising file

This file determines values of Ising interactions J_{ij}^z (for $S = 1/2$ system). For fermion electronic system, Ising terms are given as

$$H+ = \sum_{i,j} J_{ij}^z (n_{i\uparrow} - n_{i\downarrow})(n_{j\uparrow} - n_{j\downarrow}). \tag{4.12}$$

For Spin system, they are given as

$$H+ = \sum_{i,j} J_{ij}^z S_i^z S_j^z. \tag{4.13}$$

An example of file format is shown as follows.

```
======================
NIsing 6
======================
========Ising ======
======================
   0     1  0.50000
   1     2  0.50000
   2     3  0.50000
   3     4  0.50000
   4     5  0.50000
   5     0  0.50000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of Ising interactions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of Ising interactions.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index ($0 <=$ [int02], [int03] $<$ `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for J_{ij}^{z}.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of Ising interactions are double counted.
- A program is terminated, when [int01] is different from the total number of Ising interactions defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.13 PairLift file

This file determines values of PairLift couplings J_{ij}^{PairLift} (for $S = 1/2$ system),

$$H+ = \sum_{i,j} J_{ij}^{\text{PairLift}} (c_{i\uparrow}^{\dagger} c_{i\downarrow} c_{j\uparrow}^{\dagger} c_{j\downarrow} + c_{i\downarrow}^{\dagger} c_{i\uparrow} c_{j\downarrow}^{\dagger} c_{j\uparrow}). \tag{4.14}$$

An example of file format is shown as follows.

```
======================
NPairLift 6
======================
========NPairLift ======
======================
   0     1  0.50000
   1     2  0.50000
   2     3  0.50000
   3     4  0.50000
   4     5  0.50000
   5     0  0.50000
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [double01]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of PairLift couplings. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of PairLift couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int03] < `Nsite`).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for $J_{ij}^{\rm PairLift}$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of PairLift couplings are double counted.
- A program is terminated, when [int01] is different from the total number of PairLift couplings defined in this file.
- A program is terminated, when either [int02] or [int03] are out of range from the defined values.

4.2.14 OneBodyG file

This file determines the target components of one-body Green's function $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$. An example of file format is shown as follows.

```
===============================
NCisAjs         24
===============================
======== Green functions ======
===============================
    0      0      0      0
    0      1      0      1
    1      0      1      0
    1      1      1      1
    2      0      2      0
    2      1      2      1
    3      0      3      0
    3      1      3      1
    4      0      4      0
    4      1      4      1
    5      0      5      0
    5      1      5      1
    6      0      6      0
    6      1      6      1
    7      0      7      0
    7      1      7      1
    8      0      8      0
    8      1      8      1
    9      0      9      0
    9      1      9      1
   10      0     10      0
   10      1     10      1
   11      0     11      0
   11      1     11      1
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [int05]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of one-body Green's functions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of one-body Green's functions.

- [int02], [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int04] < `Nsite`).

- [int03], [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a spin index,
  0: up-spin,
  1: down-spin.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of one-body Green's functions are double counted.
- A program is terminated, when [int01] is different from the total number of one-body Green's functions defined in this file.
- A program is terminated, when [int02]-[int05] are out of range from the defined values.

4.2.15 TwoBodyG file

This file determines the target components of two-body Green's function $\langle c^\dagger_{i\sigma_1} c_{j\sigma_2} c^\dagger_{k\sigma_3} c_{l\sigma_4} \rangle$. For Spin, the condition $i = j$ and $k = l$ must be satisfied. An example of file format is shown as follows.

```
=============================================
NCisAjsCktAltDC        576
=============================================
======== Green functions for Sq AND Nq ======
=============================================
    0     0     0     0     0     0     0     0
    0     0     0     0     0     1     0     1
    0     0     0     0     1     0     1     0
    0     0     0     0     1     1     1     1
    0     0     0     0     2     0     2     0
    0     0     0     0     2     1     2     1
    0     0     0     0     3     0     3     0
    0     0     0     0     3     1     3     1
    0     0     0     0     4     0     4     0
    0     0     0     0     4     1     4     1
    0     0     0     0     5     0     5     0
    0     0     0     0     5     1     5     1
    0     0     0     0     6     0     6     0
    0     0     0     0     6     1     6     1
    0     0     0     0     7     0     7     0
    0     0     0     0     7     1     7     1
    0     0     0     0     8     0     8     0
    0     0     0     0     8     1     8     1
    0     0     0     0     9     0     9     0
    0     0     0     0     9     1     9     1
    0     0     0     0    10     0    10     0
    0     0     0     0    10     1    10     1
    0     0     0     0    11     0    11     0
    0     0     0     0    11     1    11     1
    0     1     0     1     0     0     0     0
    ...
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [int05] [int06] [int07] [int08] [int09]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of two-body Green's functions. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of two-body Green's functions.

- [int02], [int04],[int06], [int08]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int04], [int06], [int08] < `Nsite`).

- [int03], [int05],[int07], [int09]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a spin index,
  0: up-spin,
  1: down-spin.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of two-body Green's functions are double counted.
- A program is terminated, when $i = j$ and $k = l$ are not satisfied for spin model.
- A program is terminated, when [int01] is different from the total number of two-body Green's functions defined in this file.
- A program is terminated, when [int02]-[int09] are out of range from the defined values.

4.2.16 SingleExcitation file

Operators to generate single excited state $S_{i\sigma_1} c_{i\sigma_1} (S_{i\sigma_1} c_{i\sigma_1}^\dagger)$ are defined. An example of file format is shown as follows.

```
===============================
NSingle         24
===============================
======== Single Excitation ======
===============================
    0     0     0    1.0    0.0
    0     1     0    1.0    0.0
    1     0     0    1.0    0.0
   (continue...)
   11     0    0    1.0    0.0
   11     1    0    1.0    0.0
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [double01] [double02]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of single excitation operators. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of total number of single excitation operators.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02] < `Nsite`).

- [int03]

  Type : int-type (blank parameter not allowed)

Description : An integer giving a spin index,
0: up-spin,
1: down-spin.

- [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a type of single excitation operators,
  0: $c_{i\sigma_1}$,
  1: $c^{\dagger}_{i\sigma_1}$.

- [double01], [double02]

  Type : double-type (blank parameter not allowed)

  Description :[double01] gives the real part of $S_{i\sigma_1}$, while [double02] gives the imaginary part of $S_{i\sigma_1}$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of single excitation operators are double counted.
- A program is terminated, when $i = j$ and $k = l$ are not satisfied for spin model.
- A program is terminated, when [int01] is different from the total number of two-body Green's functions defined in this file.
- A program is terminated, when [int02]-[int04] are out of range from the defined values.

4.2.17 PairExcitation file

Operators to generate pair excited state $S_{i\sigma_1 j\sigma_2} c_{i\sigma_1} c^{\dagger}_{j\sigma_2}$ ($S_{i\sigma_1 j\sigma_2} c^{\dagger}_{i\sigma_1} c_{j\sigma_2}$) are defined. The type of pair excitation operators ($c_{i\sigma_1} c^{\dagger}_{j\sigma_2}$ or $c^{\dagger}_{i\sigma_1} c_{j\sigma_2}$) must be same in the input file. In S_z conserved system, σ_1 must be equal to σ_2. An example of file format is shown as follows.

```
===============================
NPair         24
===============================
======== Pair Excitation ======
===============================
    0     0     0     0    0    1.0    0.0
    0     1     0     1    0    1.0    0.0
    1     0     1     0    0    1.0    0.0
    (continue...)
   11     0    11     0    0    1.0    0.0
   11     1    11     1    0    1.0    0.0
```

File format

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3-5: Header
- Lines 6-: [int02] [int03] [int04] [int05] [int06] [double01] [double02]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of single excitation operators. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of total number of single excitation operators.

- [int02], [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int02], [int04] < `Nsite`).

- [int03], [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a spin index,
  0: up-spin,
  1: down-spin.

- [int06]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a type of pair excitation operators,
  0: $c_{i\sigma_1} c_{j\sigma_2}^{\dagger}$,
  1: $c_{i\sigma_1}^{\dagger} c_{j\sigma_2}$.

- [double01], [double02]

  Type : double-type (blank parameter not allowed)

  Description :[double01] gives the real part of $S_{i\sigma_1 j\sigma_2}$, while [double02] gives the imaginary part of $S_{i\sigma_1 j\sigma_2}$.

Use rules

- Headers cannot be omitted.
- A program is terminated, when components of single excitation operators are double counted.
- A program is terminated, when $i = j$ and $k = l$ are not satisfied for spin model.
- A program is terminated, when [int01] is different from the total number of two-body Green's functions defined in this file.
- A program is terminated, when [int02]-[int06] are out of range from the defined values.

4.2.18 SpectrumVec File

The header of input file for the initial vector to calculate dynamical Green's functions is defined. The file name and the file format (binary type) are shown as follows.

File name

- ##_rank_$$.dat

## is a name of head indicated by the key word `SpectrumVec` in the `CalcMod` file, $$ is a number of rank respectively.

File format

- Line 1: [int01]
- Line 2: [int02]
- Lines 3 -: [double01] [double02]

Parameters

- [int01]

  Type : int-type

  Description : A total number of targets of Hilbert spaces.

- [int02]

  Type : int-type

  Description : A number of Lanczos or TPQ steps.

- [double01], [double02]

  Type : double

  Description : The coefficient value of an input vector.
  [double01] is a real part, [double02] is an imaginary part.

4.3 Output files

In this section, details of output files for expert mode are explained.

4.3.1 CHECK_Chemi.dat

This file is outputted to check the input of chemical potential $\mu_{i\sigma}$,

$$H+=\sum_{i,\sigma}\mu_{i\sigma}c_{i\sigma}^{\dagger}c_{i\sigma}. \tag{4.15}$$

An example of file format is shown as follows.

```
i=0 spin=0 isite1=1 tmp_V=0.000000
i=1 spin=0 isite1=2 tmp_V=0.000000
i=2 spin=0 isite1=3 tmp_V=0.000000
i=3 spin=0 isite1=4 tmp_V=0.000000
i=4 spin=0 isite1=5 tmp_V=0.000000
i=5 spin=0 isite1=6 tmp_V=0.000000
...
```

File format

- i=[int01] spin=[int02] isite1=[int03] tmp_V=[double01]

Parameters

- [int01]

  Type : int-type

  Description : A counted number of inputting terms.

- [int02]

  Type : int-type

  Description : An integer for showing a spin index of $\mu_{i\sigma}$,
  0: up-spin,
  1: down-spin.

- [int03]

  Type : int-type

  Description : An integer for showing a site index of $\mu_{i\sigma}$.

- [double01]

  Type : double-type

  Description : A value for $\mu_{i\sigma}$.

4.3.2 CHECK_InterAll.dat

This file is outputted to check the input of diagonal components of general two body interactions,

$$H+=\sum_{i,j,\sigma} I_{iijj\sigma_1\sigma_1\sigma_2\sigma_2} c_{i\sigma_1}^{\dagger} c_{i\sigma_1} c_{i\sigma_2}^{\dagger} c_{i\sigma_2}. \tag{4.16}$$

An example of file format is shown as follows.

```
i=0 isite1=1 A_spin=0 isite2=2 B_spin=0 tmp_V=0.500000
i=1 isite1=1 A_spin=0 isite2=2 B_spin=1 tmp_V=-0.500000
i=2 isite1=1 A_spin=1 isite2=2 B_spin=0 tmp_V=-0.500000
i=3 isite1=1 A_spin=1 isite2=2 B_spin=1 tmp_V=0.500000
i=4 isite1=2 A_spin=0 isite2=3 B_spin=0 tmp_V=0.500000
i=5 isite1=2 A_spin=0 isite2=3 B_spin=1 tmp_V=-0.500000
...
```

File format

- i=[int01] isite1=[int02] A_spin=[int03] isite2=[int04] B_spin=[int05] tmp_V=[double01]

Parameters

- [int01]

  Type : int-type

  Description : A counted number of inputting terms.

- [int02], [int04]

  Type : int-type

  Description : An integer for showing a site index of $I_{iijj\sigma_1\sigma_1\sigma_2\sigma_2}$.
  [int02] and [int04] correspond to i and j, respectively.

- [int03], [int05]

  Type : int-type

  Description : An integer for showing a spin index of $I_{iijj\sigma_1\sigma_1\sigma_2\sigma_2}$,
  0: up-spin,
  1: down-spin.
  [int03] and [int05] correspond to σ_1 and σ_2, respectively.

- [double01]

  Type : double-type

  Description : A value for $I_{iijj\sigma_1\sigma_1\sigma_2\sigma_2}$.

4.3.3 CHECK_CoulombIntra.dat

This file is outputted to check the input of on-site interactions U_i,

$$H+=\sum_i U_i n_{i\uparrow} n_{j\downarrow}. \tag{4.17}$$

An example of file format is shown as follows.

```
i=0 isite1=1 tmp_V=4.000000
i=1 isite1=2 tmp_V=4.000000
i=2 isite1=3 tmp_V=4.000000
i=3 isite1=4 tmp_V=4.000000
i=4 isite1=5 tmp_V=4.000000
i=5 isite1=6 tmp_V=4.000000
```

File format

- i=[int01] isite1=[int02] tmp_V=[double01]

Parameters

- [int01]

  Type : int-type

  Description : A counted number of inputting terms.

- [int02]

  Type : int-type

  Description : An integer for showing a site index of U_i.

- [double01]

  Type : double-type

  Description : A value for U_i.

4.3.4 CHECK_Hund.dat

This file is outputted to check the input of Hund couplings J_{ij}^{Hund},

$$H + = -\sum_{i,j} J_{ij}^{\text{Hund}} (n_{i\uparrow} n_{j\uparrow} + n_{i\downarrow} n_{j\downarrow}). \tag{4.18}$$

An example of file format is shown as follows.

```
i=0 isite1=1 isite2=2 tmp_V=0.250000
i=1 isite1=2 isite2=3 tmp_V=0.250000
i=2 isite1=3 isite2=4 tmp_V=0.250000
i=3 isite1=4 isite2=5 tmp_V=0.250000
i=4 isite1=5 isite2=6 tmp_V=0.250000
i=5 isite1=6 isite2=1 tmp_V=0.250000
```

File format

- i=[int01] isite1=[int02] isite2=[int03] tmp_V=[double01]

Parameters

- [int01]

  Type : int-type

  Description : A counted number of inputting terms.

- [int02], [int03]

  Type : int-type

  Description : An integer for showing a site index of J_{ij}^{Hund}.
  [int02] and [int03] correspond to i and j, respectively.

- [double01]

  Type : double-type

  Description : A value for J_{ij}^{Hund}.

4.3.5 CHECK_INTER_U.dat

This file is outputted to check the input of diagonal components of on-site interactions V_{ij},

$$H+=\sum_{i} V_{ij} n_i n_j \tag{4.19}$$

An example of file format is shown as follows.

```
i=0 isite1=1 isite2=2 tmp_V=-0.125000
i=1 isite1=2 isite2=3 tmp_V=-0.125000
i=2 isite1=3 isite2=4 tmp_V=-0.125000
i=3 isite1=4 isite2=5 tmp_V=-0.125000
i=4 isite1=5 isite2=6 tmp_V=-0.125000
i=5 isite1=6 isite2=1 tmp_V=-0.125000
```

File format

- i=[int01] isite1=[int02] isite2=[int03] tmp_V=[double01]

Parameters

- [int01]

  Type : int-type

  Description : A counted number of inputting terms.

- [int02], [int03]

  Type : int-type

  Description : An integer giving a site index of V_{ij}.
  [int02] and [int03] correspond to i and j, respectively.

- [double01]

  Type : double-type

  Description : A value for V_{ij}.

4.3.6 CHECK_Memory.dat

This file shows the size of memory using the calculation. An example of file format is shown as follows.

```
MAX DIMENSION idim_max=400
REQUIRED MEMORY  max_mem=0.000019 GB
```

File format

- MAX DIMENSION idim_max=[int01]
- REQUIRED MEMORY max_mem =[double01] GB

Parameters

- [int01]

  Type : int-type

  Description : An integer to show total numbers of the Hilbert space under a calculation.

- [double01]

  Type : double-type

  Description : A size of memory to store the Hilbert space under a calculation (GB unit).

4.3.7 WarningOnTransfer.dat

This file shows the components double counted of transfer integrals. An example of file format is shown as follows.

```
double conuntings in transfers: i=0 j=2 spni 0 spnj 0
double conuntings in transfers: i=2 j=0 spni 0 spnj 0
double conuntings in transfers: i=0 j=2 spni 1 spnj 1
double conuntings in transfers: i=2 j=0 spni 1 spnj 1
```

File format

- double countings in transfers: i=[int01] j=[int02] spni [int03] spnj [int04]

Parameters

- [int01], [int02]

  Type : int-type

  Description : An integer of a site number where transfer integrals are double counted.

- [int03], [int04]

  Type : int-type

  Description : An integer of a spin index of a transfer integral,
  0: up-spin,
  1: down-spin.

4.3.8 CalcTimer.dat

The name of calculation process, process number and the calculation process time are output in order at each lines in the CalcTime.dat file. An example of file format for TPQ method is shown as follows.

```
All                                                 [0000]     12.94052
  sz                                                [1000]      0.01795
  diagonalcalc                                      [2000]      0.00693
  CalcByTPQ                                         [3000]     12.90670
    FirstMultiply                                   [3100]      0.08416
      rand   in FirstMultiply                       [3101]      0.00172
      mltply in FirstMultiply                       [3102]      0.07707
    expec_energy_flct                               [3200]      9.06255
      calc flctuation in expec_energy_flct          [3201]      1.67779
      mltply in expec_energy_flct                   [3202]      7.31207
    expec_onebody                                   [3300]      0.11640
    expec_twobody                                   [3400]      3.28796
    Multiply                                        [3500]      0.14840
    FileIO                                          [3600]      0.20493
================================================
All mltply                                          [0001]      7.38883
  diagonal                                          [0100]      0.04153
  Hubbard                                           [0300]      7.34636
    trans    in Hubbard                             [0310]      7.34595
      double                                        [0311]      0.00000
      single                                        [0312]      0.00000
      inner                                         [0313]      7.34299
    interall in Hubbard                             [0320]      0.00008
      interPE                                       [0321]      0.00000
      inner                                         [0322]      0.00000
    pairhopp in Hubbard                             [0330]      0.00006
      interPE                                       [0331]      0.00000
      inner                                         [0332]      0.00000
    exchange in Hubbard                             [0340]      0.00004
      interPE                                       [0341]      0.00000
      inner                                         [0342]      0.00000
================================================
```

4.3.9 TimeKeeper.dat

This file is outputted to show the calculation process information. An example of file format for Lanczos method is shown as follows.

```
diagonal calculation finishes: Wed Sep 16 22:58:49 2015
Lanczos Eigen Value start: Wed Sep 16 22:58:49 2015
1 th Lanczos step: Wed Sep 16 22:58:49 2015
  ...
122 th Lanczos step: Wed Sep 16 22:58:49 2015
Lanczos Eigenvalue finishes: Wed Sep 16 22:58:49 2015
Lanczos Eigenvector finishes: Wed Sep 16 22:58:49 2015
Lanczos expec energy finishes: Wed Sep 16 22:58:49 2015
CG Eigenvector finishes: Wed Sep 16 22:58:49 2015
CG expec energy finishes: Wed Sep 16 22:58:50 2015
CG expec_cisajs finishes: Wed Sep 16 22:58:50 2015
CG expec_cisajacktalt begins: Wed Sep 16 22:58:50 2015
```

File name

- ##_TimeKeeper.dat

## indicates a header defined by [string02] in a ModPara file.

4.3.10 sz_TimeKeeper.dat

This file is outputted to show the process information to obtain the Hilbert space needed for calculation . An example of file format is shown as follows.

```
initial sz : Wed Sep 16 22:58:49 2015
num_threads==4
omp parallel sz finishes: Wed Sep 16 22:58:49 2015
mid omp parallel sz : Wed Sep 16 22:58:49 2015
omp parallel sz finishes: Wed Sep 16 22:58:49 2015
```

File name

- ##_sz_TimeKeeper.dat

## indicates a header defined by [string02] in a ModPara file.

4.3.11 Time_CG_EigenVector.dat

(For Lanczos method) The process for calculating eigenvector by CG method is outputted. An example of file format is shown as follows.

```
allocate succeed !!!
b[4341]=1.000000 bnorm== 1.000000
i_itr=0 itr=5 0.0411202543 0.0000100000
...
i_itr=0 itr=155 0.0000000058 0.0000100000
CG OK:   t_itr=155
i_itr=0 itr=155 time=0.000000
fabs(fabs(xb)-1.0)=0.9955114473313577
b[4341]=0.004489 bnorm== 1.000000
i_itr=1 itr=5 13.0033983157 0.0000100000
...
CG OK:   t_itr=275
i_itr=1 itr=120 time=0.000000
fabs(fabs(xb)-1.0)=0.0000000000001295
number of iterations in inv1:i_itr=1 itr=120
t_itr=275 0.000000
```

File name

- ##_Time_CG_EigenVector.dat

## indicates a header defined by [string02] in a ModPara file.

4.3.12 energy.dat

(For Lanczos method) The values of energy, doublon and $\langle S_z \rangle$ calculated by using eigenvector obtained by Lanczos or CG method are outputted. An example of file format is shown as follows.

```
Energy  -7.1043675920
Doublon  0.4164356536
Sz  0.0000000000
```

File name

- ##_energy.dat

## indicates a header defined by [string02] in a ModPara file.

File format

- Line 1: Energy [double01]
- Line 2: Doublon [double02]
- Line 3: Sz [double02]

Parameters

- [double01]

  Type : double-type

  Description : The value of energy calculated by the eigenvetor obtained by Lanczos or CG method.

- [double02]

  Type : double-type

  Description : The value of doublon calculated by the eigenvetor obtained by Lanczos or CG method, $\frac{1}{N_s}\sum_i \langle n_{i\uparrow} n_{i\downarrow} \rangle$ (N_s is a total number of sites).

- [double03]

  Type : double-type

  Description : The value of S_z calculated by the eigenvetor obtained by Lanczos or CG method.

4.3.13 Lanczos_Step.dat

(For Lanczos method) This file is outputted to show the process information for calculating eigenvector by Lanczos method. An example of file format for Lanczos method is shown as follows.

```
stp=4 -4.7732163164 -1.7790936582 1.2246691944 4.4823068739
stp=6 -5.8159638812 -3.6807425834 -1.3738652472 0.9326262962
stp=8 -6.2772935672 -4.8650501184 -3.1096996066 -1.1940653342
...
stp=120 -7.1043675920 -7.0817672578 -7.0646589929 -7.0008766356
stp=122 -7.1043675920 -7.0817672578 -7.0646589929 -7.0008766356
```

File name

- ##_Lanczos_Step.dat

## indicates a header defined by [string02] in a ModPara file.

File format

- stp= [int01] [double01] [double02] [double03] [double04]

Parameters

- [int01]

  Type : int-type

  Description : An integer showing Lanczos step.

- [double01], [double02], [double03], [double04]

  Type : double-type

  Description : The eigenvalue at the Lanczos step [int01].
  [double01] is a ground energy, [double02], [double03] and [double04] are 1st, 2nd and 3rd excited energies, respectively.

4.3.14 Time_TPQ_Step.dat

(For TPQ method) This file is outputted to show the time for starting calculation of TPQ method at each seeds and steps. An example is shown as follows.

```
set 0 step 1:TPQ begins: Wed Jul 13 07:59:20 2016
set 0 step 2:TPQ begins: Wed Jul 13 07:59:20 2016
set 0 step 3:TPQ begins: Wed Jul 13 07:59:20 2016
...
set 4 step 1997:TPQ begins: Wed Jul 13 07:59:32 2016
set 4 step 1998:TPQ begins: Wed Jul 13 07:59:32 2016
set 4 step 1999:TPQ begins: Wed Jul 13 07:59:32 2016
```

File name

- ##_TPQ_Step.dat

## indicates a header defined by [string02] in a ModPara file.

File format

- set [int01] stp [int02]: TPQ begins: [string01]

Parameters

- [int01]

  Type : int-type

  Description : A seed number under calculation of TPQ method.

- [int02]

  Type : int-type

  Description : A step number under calculation of TPQ method.

- [string01]

  Type : string-type

  Description : A time for starting calculation of TPQ method at each seeds and steps.

4.3.15 Norm_rand.dat

(For TPQ method) This file is outputted to show the calculation process information for TPQ method. An example of file format is shown as follows.

```
 # inv_temp, global_norm, global_1st_norm, step_i
0.017471 19.046586 11.288975 1
0.034863 19.089752 11.288975 2
...
31.999572 20.802362 11.288975 1997
32.015596 20.802362 11.288975 1998
32.031620 20.802362 11.288975 1999
```

File name

- Norm_rand??.dat

?? indicates a number of runs under calculation of TPQ method.

File format

- Line 1: Header
- Lines 2-: [double01] [double02] [double03] [int01]

Parameters

- [double01]

  Type : double-type

  Description : Inverse temperature $1/k_{\rm B}T$.

- [double02]

  Type : double-type

  Description : A norm of a wave function before normalization given by $\langle\tilde{\psi}_k|\tilde{\psi}_k\rangle$, where $|\tilde{\psi}_k\rangle \equiv (l - \hat{H}/N_s)|\psi_{k-1}\rangle$.

- [double03]

  Type : double-type

  Description : A norm of an initial wave function before normalization given by $\langle\tilde{\psi}_0|\tilde{\psi}_0\rangle$, where $|\tilde{\psi}_0\rangle$ is an initial random vector.

- [int01]

  Type : int-type

  Description : A number of operations of $(l - \hat{H}/N_s)$ to an initial wave function, where l is `LargeValue` defined in ModPara file and N_s is a total number of sites.

4.3.16 SS_rand.dat

(For TPQ method) This file is outputted to show the calculation results for TPQ method. An example of file format is shown as follows.

```
# inv_tmp, energy, phys_var, phys_doublon, phys_num, step_i
0.017471  5.526334 45.390269 1.464589 6.000000 1
0.034863  5.266718 42.655559 1.434679 6.000000 2
...
31.999572  -4.814170 23.176231 0.590568 6.000000 1997
32.015596  -4.814170 23.176231 0.590568 6.000000 1998
32.031620  -4.814170 23.176231 0.590568 6.000000 1999
```

File name

- SS_rand??.dat

?? indicates a number of runs under calculation of TPQ method.

File format

- Line 1: Header
- Lines 2-: [double01] [double02] [double03] [double04] [double05] [int01]

Parameters

- [double01]

  Type : double-type

  Description : Inverse temperature $1/k_{\rm B}T$.

- [double02]

  Type : double-type

  Description : The expected value of energy $\langle H \rangle$.

- [double03]

  Type : double-type

  Description : The expected value of square of Hamiltonian $\langle H^2 \rangle$.

- [double03]

  Type : double-type

  Description : The expected value of doublon, $\frac{1}{N_s}\sum_i \langle n_{i\uparrow} n_{i\downarrow} \rangle$ (N_s is a total number of sites).

- [double05]

  Type : double-type

  Description : A total number of particles $\langle \hat{n} \rangle$.

- [int01]

  Type : int-type

  Description : A number of operations of $(l - \hat{H}/N_s)$ to an initial wave function, where l is `LargeValue` defined in ModPara file and N_s is a total number of sites.

4.3.17 Flct_rand.dat

(For TPQ method) This file is outputted to show the calculation results of the fluctuation of the particle number, doublon and S_z for TPQ method. An example of file format is shown as follows.

```
 # inv_temp, N, N^2, D, D^2, Sz, Sz^2, step_i
0.0826564 12.00 144.00 0.00 0.00 0.0009345626081113 0.2500 1
0.1639935 12.00 144.00 0.00 0.00 0.0023147006319775 0.2500 2
0.2440168 12.00 144.00 0.00 0.00 0.0037424057659867 0.2500 3
...
135.97669 12.00 144.00 0.00 0.00 -0.0000000000167368 0.2500 1998
136.04474 12.00 144.00 0.00 0.00 -0.0000000000165344 0.2500 1999
```

File name

- Flct_rand??.dat

?? indicates a number of runs under calculation of TPQ method.

File format

- Line 1: Header
- Lines 2-: [double01] [double02] [double03] [double04] [double05] [double06] [double07] [int01]

Parameters

- [double01]

  Type : double-type

  Description : Inverse temperature $1/k_{\mathrm{B}}T$.

- [double02]

  Type : double-type

  Description : A total particle number $\sum_i \langle \hat{n}_i \rangle$.

- [double03]

  Type : double-type

  Description : The expected value of square of particle number $\sum_i \langle \hat{n}_i^2 \rangle$.

- [double04]

  Type : double-type

  Description : The expected value of doublon $\frac{1}{N_s} \sum_i \langle n_{i\uparrow} n_{i\downarrow} \rangle$ (N_s is a total number of sites).

- [double05]

  Type : double-type

  Description : The expected value of square of doublon $\frac{1}{N_s}\sum_i \langle (n_{i\uparrow}n_{i\downarrow})^2\rangle$ (N_s is a total number of sites).

- [double06]

  Type : double-type

  Description : The expected value of S_z $\frac{1}{N_s}\sum_i \langle \hat{S}_i^z\rangle$ (N_s is a total number of sites).

- [double07]

  Type : double-type

  Description : The expected value of square of S_z $\frac{1}{N_s}\sum_i \langle (\hat{S}_i^z)^2\rangle$ (N_s is a total number of sites).

- [int01]

  Type : int-type

  Description : A number of operations of $(l - \hat{H}/N_s)$ to an initial wave function, where l is `LargeValue` defined in ModPara file and N_s is a total number of sites.

4.3.18 Eigenvalue.dat

(For FullDiag method) This file is outputted to show the energies calculated by FullDiag method. An example of file format is shown as follows.

```
0 -4.8141698096
1 -3.7968502453
2 -3.2462822372
...
397 13.9898305290
398 14.4896221034
399 14.8525199079
```

File format

- [int01] [double01]

Parameters

- [int01]

  Type : int-type

  Description : An index of eigenvalues. The index 0 is for an energy of a ground state and indexes are counted from the low order of energies.

- [double01]

  Type : double-type

  Description : The expected value of energy $\langle H \rangle$.

4.3.19 phys.dat

(For FullDiag method) This file is outputted to show the physical values calculated by FullDiag method. The datas are outputted in the low order of energies. An example of file format is shown as follows.

```
<H>          <N>          <Sz>         <S2>         <D>
 -4.814170    0.000000    0.000000   -0.000000    0.590568
 -3.796850    0.000000    0.000000    1.333333    0.423804
...
14.489622    0.000000    0.000000    0.000000    2.550240
14.852520    0.000000    0.000000    0.000000    2.329157
```

File name

- Canonical ensemble: ##_phys_Nup_$$Ndown%%.dat
- Grand canonical ensemble: ##_phys.dat

##, $$ and %% indicate [string02], Nup and Ndown defined in a ModPara file, respectively.

File format

- Line 1: Header
- Lines 2-: [double01] [double02] [double03] [double04] [double05]

Parameters

- [double01]

  Type : double-type

  Description : The energy $\langle H \rangle$.

- [double02]

  Type : double-type

  Description : A total number of particles $\langle \hat{n} \rangle$.

- [double03]

  Type : double-type

  Description : The expected value of S_z, $\langle S_z \rangle$.

- [double04]

  Type : double-type

  Description : The expected value of $\boldsymbol{S}^2$, $\langle \boldsymbol{S}^2 \rangle$.

- [double05]

  Type : double-type

  Description : The expected value of doublon, $\frac{1}{N_s}\sum_i \langle n_{i\uparrow} n_{i\downarrow}\rangle$ (N_s is a total number of sites).

4.3.20 cisajs.dat

This file is output files for one-body Green's function $\langle c^\dagger_{i\sigma_1} c_{j\sigma_2}\rangle$. The target components are set in the input file with the keyword "OneBodyG". An example of file format is shown as follows.

```
    0    0    0    0 0.4452776740 0.0000000000
    0    1    0    1 0.4452776740 0.0000000000
    1    0    1    0 0.5000000000 0.0000000000
    1    1    1    1 0.5000000000 0.0000000000
    2    0    2    0 0.4452776740 0.0000000000
    2    1    2    1 0.4452776740 0.0000000000
    3    0    3    0 0.5000000000 0.0000000000
    3    1    3    1 0.5000000000 0.0000000000
    ...
```

File name

- Lanczos method: ##_cisajs.dat
- TPQ method: ##_cisajs_set??step%%.dat
- Full diagonalization method: ##_cisajs_eigen&&.dat

##, ??, %% and && indicate [string02] in ModPara file, a number of runs under calculation in TPQ method, a number of steps in TPQ method and an index of eigenvalues, respectively.

File format

- [int01] [int02] [int03] [int04] [double01] [double02]

Parameters

- [int01], [int03]

  Type : int-type

  Description : An integer of the site number. [int01] and [int03] show the i and j site numbers, respectively.

- [int02], [int04]

  Type : int-type

  Description : An integer of the spin index,
  0: up-spin,
  1: down-spin.
  [int02] and [int04] show σ_1 and σ_2, respectively.

- [double01], [double02]

  Type : double-type

  Description : The value of $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$.
  [double01] and [double02] show the real and imaginary part of $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$, respectively.

4.3.21 cisajscktalt.dat

This file is output files for Two body Green's function $\langle c^\dagger_{i\sigma_1} c_{j\sigma_2} c^\dagger_{k\sigma_3} c_{l\sigma_4} \rangle$. The target components are set in the input file with the keyword "TwoBodyG". An example of file format is shown as follows.

```
    0    0    0    0    0    0    0    0 0.4452776740 0.0000000000
    0    0    0    0    0    1    0    1 0.1843355815 0.0000000000
    0    0    0    0    1    0    1    0 0.1812412105 0.0000000000
    0    0    0    0    1    1    1    1 0.2640364635 0.0000000000
    0    0    0    0    2    0    2    0 0.0279690007 0.0000000000
    0    0    0    0    2    1    2    1 0.2009271524 0.0000000000
    0    0    0    0    3    0    3    0 0.2512810778 0.0000000000
    0    0    0    0    3    1    3    1 0.1939965962 0.0000000000
    ...
```

File name

- Lanczos method: ##_cisajscktalt.dat
- TPQ method: ##_cisajscktalt_set??step%%.dat
- Full diagonalization method: ##_cisajscktalt_eigen&&.dat

##, ??, %% and && indicate [string02] in ModPara file, a number of runs under calculation in TPQ method, a number of steps in TPQ method and an index of eigenvalues, respectively.

File format

- [int01] [int02] [int03] [int04] [int05] [int06] [int07] [int08] [double01] [double02]

Parameters

- [int01], [int03],[int05], [int07]

  Type : int-type

  Description : An integer of the site number. [int01], [int03], [int05] and [int07] show the i, j, k and $l$site numbers, respectively.

- [int02], [int04],[int06], [int08]

  Type : int-type

  Description : An integer of the spin index,
  0: up-spin,
  1: down-spin.
  [int02], [int04], [int06] and [int08] show σ_1, σ_2, σ_3 and σ_4, respectively.

- [double01], [double02]

  Type : double-type

  Description : The value of $\langle c^{\dagger}_{i\sigma_1} c_{j\sigma_2} c^{\dagger}_{k\sigma_3} c_{l\sigma_4}\rangle$.
  [double01] and [double02] show the real and imaginary part of $\langle c^{\dagger}_{i\sigma_1} c_{j\sigma_2} c^{\dagger}_{k\sigma_3} c_{l\sigma_4}\rangle$, respectively.

4.3.22 eigenvec.dat

When OutputEigenVec=1 in a CalcMod file, eigenvectors calculated by a Lanczos method are outputted. When InputEigenVec=1 in a CalcMod file, eigenvectors are inputted by this outputted file. A file format is a binary type.

File name

- ##_eigenvec_&&_rank_$$.dat

## indicates [string02] in ModPara file, && is a number of eigenvalues, $$ is a number of rank respectively.

File format

- Line 1: [int01]
- Line 2: [int02]
- Lines 3 -: [double01] [double02]

Parameters

- [int01]

  Type : int-type

  Description : A total number of targets of Hilbert spaces.

- [int02]

  Type : int-type

  Description : A number of Lanczos steps.

- [double01], [double02]

  Type : double

  Description : A value of eigenvectors.
  [double01] is a real part, [double02] is an imaginary part.

4.3.23 tmpvec.dat

When ReStart=1, 2 in a CalcMod file, vectors after stopping calculation at an indicated step are outputted. A file format is a binary type. An example of file format is shown as follows.

File name

- Lanczos method: ##_tmpvec_rank_$$.dat
- TPQ method: ##_tmpvec_set_&&_rank_$$.dat

## indicates [string02] in ModPara file, $$ is a number of rank respectively. && is a sampling number for TPQ calculation.

File format

- Line 1: [int01]
- Line 2: [int02]
- Lines 3 - 3+[int01]: [double01] [double02]
- Lines 4+[int01] - 3+2×[int01]: [double03] [double04]

Parameters

- [int01]

  Type : int-type

  Description : A total number of targets of Hilbert spaces.

- [int02]

  Type : int-type

  Description : A number of Lanczos (TPQ) steps for Lanczos (TPQ) method.

- [double01], [double02]

  Type : double

  Description : A value of two vectors ($\boldsymbol{v}_{k+1}$ in Eq. (5.6) for Lanczos method and $\boldsymbol{y}_{k+1}$ in Eq .(5.8) for TPQ method). [double01] is a real part, [double02] is an imaginary part.

- [double03], [double04]

  Type : double

  Description : A value of two vectors ($\boldsymbol{v}_k$ in Eq. (5.6) for Lanczos method and $\boldsymbol{y}_k$ in Eq .(5.8) for TPQ method). [double03] is a real part, [double04] is an imaginary part.

4.3.24 DynamicalGreen.dat

This file is output files for calculating dynamical Green's function. An example of file format is shown as follows.

File name

- ##_DynamicalGreen.dat

## indicates [string02] in ModPara file.

File format

- Lines 1-: [double01] [double02] [double03] [double04]

Parameters

- [double01], [double02]

  Type : double-type

  Description : The value of the frequency z.
  [double01] and [double02] are a real and an imaginary part of z, respectively.

- [double03], [double04]

  Type : double-type

  Description: The value of dynamical Green's functions $G(z)$.
  [double03] and [double04] are a real and an imaginary part of $G(z)$, respectively.

4.3.25 recalcvec.dat

This file is output files for two vectors to recalculate dynamical Green's function by Lanczos method. A file format is a binary type. An example of file format is shown as follows.

File name

- ##_recalcvec_rank_$$.dat

## indicates [string02] in ModPara file, $$ is a number of rank respectively.

File format

- Line 1: [int01]
- Line 2: [int02]
- Lines 3 - 3+[int02]: [double01] [double02]
- Lines 4+[int02] - 3+2×[int02]: [double03] [double04]

Parameters

- [int01]

  Type : int-type

  Description : The step for calculating dynamical Green's functions by Lanczos method N_d.

- [int02]

  Type : int-type

  Description : A total number of targets of Hilbert spaces.

- [double01], [double02]

  Type : double-type

  Description : The value of the vector $\boldsymbol{v}_{k+1}$ for recalculating dynamical Green's functions by Lanczos method.
  [double01] and [double02] are a real part and an imaginary part of $\boldsymbol{v}_{k+1}$, respectively.

- [double03], [double04]

  Type : double-type

  Description : The value of the vector $\boldsymbol{v}_k$ for recalculating dynamical Green's functions by Lanczos method.
  [double03] and [double04] are a real part and an imaginary part of $\boldsymbol{v}_k$, respectively.

4.3.26 TMcomponents.dat

This file is output files for the components of tridiagonal matrix and the norm of the excited state to recalculate dynamical Green's function by Lanczos method. A file format is a binary type. An example of file format is shown as follows.

File name

- ##_TMcomponents.dat

## indicates [string02] in ModPara file, $$ is a number of rank respectively.

File format

- Line 1: [int01]
- Line 2: [double01]
- Lines 3 -: [double02] [double03]

Parametes

- [int01]

  Type : int-type

  Description : The step for calculating dynamical Green's functions by Lanczos method N_d.

- [double01]

  Type : double-type

  Description : The value of the norm of the excited state.

- [double02], [double03]

  Type : double-type

  Description : The value of the components of the tridiagonal matrix to recalculate dynamical Green'S functions by Lanczos method $\alpha_i,\ \beta_i\ (i = 1, \cdots N_d)$. [double02] is α_i and [double03] is β_i, respectively

4.4 Error messages

- `ERROR ! Unsupported Keyword !`

  The program stops because unsupported keyword is specified.

- `"ERROR !  Keyword` *keyword* `is duplicated !`

  The program stops because a parameter is specified twice.

- `ERROR ! Unsupported Solver :` *solver*
- `ERROR ! Unsupported Model :` *model*
- `Sorry, this system is unsupported in the STANDARD MODE...`
  `Please use the EXPART MODE, or write a NEW FUNCTION and post it us.`

  The program stops because unsupported parameter for `method`, `model`, or `lattice` is specified.

- `ERROR ! abs(2 * Sz) > nsite in Hubbard model !`
- `ERROR ! Nelec > 2 * nsite in Hubbard model !`
- `ERROR ! (nelec + 2 * Sz) % 2 != 0 in Hubbard model !`
- `ERROR ! nelec <= nsite && 2 * |Sz| > nelec in Hubbard model !`
- `ERROR ! nelec > nsite && 2 * |Sz| > 2 * nsite - nelec in Hubbard model !`
- `ERROR ! abs(2 * Sz) > nsite in Spin model !`
- `ERROR ! (nsite + 2 * Sz) % 2 != 0 in Spin model !`
- `ERROR ! abs(2 * Sz) > nsite in Hubbard model !`
- `ERROR ! Nelec_cond / 2 + Nelec_loc > nsite in Kondo model !`
- `ERROR ! (nelec_cond + nelec_loc + 2 * Sz) % 2 != 0 in Kondo model !`
- `ERROR ! nelec_cond <= nsite / 2 && 2 * |Sz| > nelec_cond + nelec_loc ...`
- `ERROR ! nelec_cond > nsite / 2 && abs(Sz2) > nsite / 2 * 3 - nelec...`

  In the calculation of the canonical ensemble, there are some irrelevant combinations of the number of electrons, the number of sites, and the total spin moment (the number of electrons is larger twice than the number of sites); If these situations are detected, the program will stop.

- `Check !` *keyword* `is SPECIFIED but will NOT be USED.`
  `        Please COMMENT-OUT this line`
  `        or check this input is REALLY APPROPRIATE for your purpose !`

  Because an unnecessary parameter is specified, the program suggests checking the input file. If that parameter is actually unnecessary, please delete or comment out this line.

- `ERROR !` *keyword* `is NOT specified !`

  The program stops because a prerequisite keyword is not specified.

- *keyword* = *value* `######  DEFAULT VALUE IS USED  ######`

  This is not an error message. The program states that the default value is used because this keyword is not specified.

5 Algorithm

5.1 Lanczos method

5.1.1 Details of Lanczos method

Some parts of this section are based on the manual of titpack [7] and textbook by M. Sugihara and K. Murota [8](These references are written in Japanese).

In the Lanczos method, by successively operating the Hamiltonian to the initial vector, we obtain the accurate eigenvalues around the maximum and minimum eigenvalues and associated eigenvectors. Because we can perform Lanczos method by using only two vectors whose dimensions are the dimension of the total Hilbert space [*1], Lanczos method is frequently used for the diagonalization of the large matrices. As we detail below, one additional vector is necessary for obtaining the eigenvector.

The principle of the Lanczos method is based on the power method. In the power method, by successively operating the Hamiltonian $\hat{\mathcal{H}}$ to the arbitrary vector $\boldsymbol{x}_0$, we generate $\hat{\mathcal{H}}^n\boldsymbol{x}_0$. The obtained space $\mathcal{K}_{n+1}(\hat{\mathcal{H}},\boldsymbol{x}_0)=\{\boldsymbol{x}_0,\hat{\mathcal{H}}^1\boldsymbol{x}_0,\dots,\hat{\mathcal{H}}^n\boldsymbol{x}_0\}$ is called Krylov subspace. Initial vector is represented by the superposition of the eigenvectors $\boldsymbol{e}_i$ (corresponding eigenvalues are E_i) of $\hat{\mathcal{H}}$ as

$$\boldsymbol{x}_0=\sum_i a_i\boldsymbol{e}_i. \tag{5.1}$$

Here, E_0 is maximum absolute values of the eigenvalues. We note that all the eigenvalues are real number because Hamiltonian is Hermite. By operating $\hat{\mathcal{H}}^n$ to the initial vector, we obtain the relation as

$$\hat{\mathcal{H}}^n\boldsymbol{x}_0=E_0^n\Big[a_0\boldsymbol{e}_0+\sum_{i\neq0}\left(\frac{E_i}{E_0}\right)^n a_i\boldsymbol{e}_i\Big]. \tag{5.2}$$

This relation indicates that the eigenvector of E_0 becomes dominant for sufficiently large n. In the Lanczos method, we obtain the eigenvalues and eigenvectors by performing the proper transformation for obtained Krylov subspace.

In the Lanczos method, we successively generate the normalized orthogonal basis $\boldsymbol{v}_0,\dots,\boldsymbol{v}_{n-1}$ from the Krylov subspace $\mathcal{K}_n(\hat{\mathcal{H}},\boldsymbol{x}_0)$. We defines initial vector and

[*1] In $\mathcal{H}\Phi$, to reduce the numerical cost, we use some additional vectors; vector for accumulating the real-space diagonal elements of the Hamiltonian, vector for specifying the given S_z space and given particle space. The dimension of these vectors is that of the Hilbert space.

associated components as $\boldsymbol{v}_0 = \boldsymbol{x}_0/|\boldsymbol{x}_0|, \beta_0 = 0, \boldsymbol{x}_{-1} = 0$. From this initial condition, we can obtain the normalized orthogonal basis as follows:

$$\alpha_k = (\hat{\mathcal{H}}\boldsymbol{v}_k, \boldsymbol{v}_k), \tag{5.3}$$

$$\boldsymbol{w} = \hat{\mathcal{H}}\boldsymbol{v}_k - \beta_k\boldsymbol{v}_{k-1} - \alpha_k\boldsymbol{v}_k, \tag{5.4}$$

$$\beta_{k+1} = |\boldsymbol{w}|, \tag{5.5}$$

$$\boldsymbol{v}_{k+1} = \frac{\boldsymbol{v}_k}{|\boldsymbol{v}_k|}. \tag{5.6}$$

From these definitions, it it obvious that α_k, β_k are real number.

In the subspace spanned by these normalized orthogonal basis, the Hamiltonian is transformed as

$$T_n = V_n^\dagger \hat{\mathcal{H}} V_n. \tag{5.7}$$

Here, V_n is matrix whose column vectors are $\boldsymbol{v}_i (i = 0, 1, \ldots, n-1)$. T_n is tridiagonal matrix and its diagonal elements are α_i and subdiagonal elements are β_i. It is known that the eigenvalues of $\hat{\mathcal{H}}$ are well approximated by the eigenvalues of T_n for sufficiently large n. (We note that $V^\dagger V = I$,I is identity matrix). The original eigenvectors of $\hat{\mathcal{H}}$ is obtained by $\boldsymbol{e}_i = V\tilde{\boldsymbol{e}}_i$, where $\tilde{\boldsymbol{e}}_i$ are the eigenvectors of T_n. From V, we can obtain the eigenvectors of $\hat{\mathcal{H}}$ by performing the Lanczos method. However, in the actual calculations, it is difficult to keep V because its dimension is large [dimension of V = (dimension of the total Hilbert space) $\times$ (# of Lanczos iterations)]. Thus, to obtain the eigenvectors, we again perform the same Lanczos calculations after we obtain the eigenvalues from the Lanczos methods. In the first Lanczos calculation, we keep $\tilde{\boldsymbol{e}}_i$ because its dimension is small [*2]. From this procedure, we obtain the eigenvectors from V.

In the Lanczos method, within a few hundred or thousand Lanczos iterations, we obtain the accurate eigenvalues near the maximum and minimum values of eigenvalues. The necessary number of iterations is small enough compared to the dimensions of the total Hilbert space. We note that it is shown that the errors of the maximum and minimum eigenvalues becomes exponentially small as a function of Lanczos iteration (for details, see Ref. [8]).

5.1.2 Inverse iteration method

From the approximate value of the eigenvalues (E_n), by successively operating $(\hat{\mathcal{H}} - E_n)^{-1}$ to the initial vector $\boldsymbol{y}_0$, we can obtain the accurate eigenvector for E_n.

From $(\hat{\mathcal{H}} - E_n)^{-1}\boldsymbol{y}_0$, we obtain the linear simultaneous equations such as

$$\boldsymbol{y}_k = (\hat{\mathcal{H}} - E_n)\boldsymbol{y}_{k+1}. \tag{5.8}$$

By solving this equation by using the conjugate gradient method (CG method), we obtain the eigenvector. From the obtained eigenvector, we can calculate the eigenvalues and correlation functions. We note that additional four vectors are necessary to perform the CG method. For large system size, it may be impossible to allocate memory to the additional vectors.

[*2]upper bound of the dimensions of $\tilde{\boldsymbol{e}}_i$ is # of Lanczos iterations.

5.1.3 Details of implementation

Initial vector

For Lanczos method, an initial vector is specified with `initial_iv`($\equiv r_s$) defined in an input file for Standard mode or a ModPara file for Expert mode. A type of an initial vector can be selected from a real number or complex number by using `InitialVecType` in a ModPara file.

- For canonical ensemble and `initial_iv` ≥ 0

  A component of a target of Hilbert space is given by

  $$(N_{\rm dim}/2 + r_s)\% N_{\rm dim}, \tag{5.9}$$

  where $N_{\rm dim}$ is a total number of the Hilbert space and $N_{\rm dim}/2$ is added to avoid selecting a special Hilbert space for a default value `initial_iv` $= 1$. When a type of an initial vector is selected as a real number, a coefficient value is given by 1, while as a complex number, the value is given by $(1+i)/\sqrt{2}$.

- For grand canonical ensemble or `initial_iv` < 0

  An initial vector is given by using a random generator, i.e. coefficients of all components for the initial vector is given by random numbers. The seed is calculated as

  $$123432 + |r_s|, \tag{5.10}$$

  where r_s is a number given by an input file and $n_{\rm run}$ is a number of runs. A maximum value of $n_{\rm run}$ is defined by `NumAve` in an input file for Standard mode or a ModPara file for Expert mode. Random numbers are generated by using SIMD-oriented Fast Mersenne Twister (dSFMT) [9].

Convergence condition

In $\mathcal{H}\Phi$, we use `dsyev` (routine of lapack) for diagonalization of T_n. We use the energy of the first excited state of T_n as a criteria of convergence. In the standard setting, after five Lanczos step, we diagonalize T_n every two Lanczos step. If the energy of the first excited states coincides with the previous energy within the specified accuracy, the Lanczos iteration finishes. The accuracy of the convergence can be specified by `CDataFileHead` (ModPara file in the expert mode).

After obtaining the eigenvalues, we again perform the Lanczos iteration to obtain the eigenvector. From the eigenvectors $|n\rangle$, we calculate energy $E_n = \langle n|\hat{\mathcal{H}}|n\rangle$ and variance $\Delta = \langle n|\hat{\mathcal{H}}^2|n\rangle - (\langle n|\hat{\mathcal{H}}|n\rangle)^2$. If E_n coincides with the eigenvalues obtained by the Lanczos iteration and Δ is smaller than the specified value, we finish diagonalization.

If the accuracy of Lanczos method is not enough, we perform the CG method to obtain the eigenvector. As an initial vector of the CG method, we use the eigenvectors obtained by the Lanczos method in the standard setting. This often accelerates the convergence.

5.2 Full Diagonalization method

5.2.1 Over view

We generate matrix of $\hat{H}$ by using the real space configuration $|\psi_j\rangle (j = 1 \cdots d_{\rm H}$, $d_{\rm H}$ is dimension of the Hilbert space): $H_{ij} = \langle\psi_i|\hat{H}|\psi_j\rangle$. By diagonalizing this matrix, we can obtain all the eigenvalues E_i and eigenvectors $|\Phi_i\rangle$ $(i = 1 \cdots d_{\rm H})$. In the diagonalization, we use lapack routine such as `dsyev` or `zheev`. We also calculate and out put the expectation values $\langle A_i\rangle \equiv \langle\Phi_i|\hat{A}|\Phi_i\rangle$. These values are used for the finite-temperature calculations.

5.2.2 Finite-temperature calculations

From $\langle A_i\rangle \equiv \langle\Phi_i|\hat{A}|\Phi_i\rangle$, we calculate finite-temperature properties by using the relation

$$\langle\hat{A}\rangle = \frac{\sum_{i=1}^{N}\langle A_i\rangle {\rm e}^{-\beta E_i}}{\sum_{i=1}^{N} {\rm e}^{-\beta E_i}}. \tag{5.11}$$

In the actual calculation are performed as the post scripts.

5.3 Finite-temperature calculations by TPQ method

Sugiura and Shimizu show that it is possible to calculate the finite-temperature properties from a few wavefunctions (in the thermodynamic limit, only one wave function is necessary) [5]. The wavefunction is called thermal pure quantum (TPQ) state. Because TPQ state can be generated by operating the Hamiltonian to the random initial wavefunction, we directly use the routine Lanczos method to the TPQ calculations. Here, we explain how to construct micro canonical TPQ (mTPQ) state, which offers the simplest way for finite-temperature calculations.

Let $|\psi_0\rangle$ a random initial vector. By operating $(l - \hat{H}/N_s)^k (l$ is constant, N_s represents number of sites) to $|\psi_0\rangle$, we obtain the kth TPQ states as

$$|\psi_k\rangle \equiv \frac{(l - \hat{H}/N_s)|\psi_{k-1}\rangle}{|(l - \hat{H}/N_s)|\psi_{k-1}\rangle|}. \tag{5.12}$$

From $|\psi_k\rangle$, we estimate corresponding inverse temperature β_k as

$$\beta_k \sim \frac{2k/N_s}{l - u_k}, \quad u_k = \langle\psi_k|\hat{H}|\psi_k\rangle/N_s, \tag{5.13}$$

where u_k is the internal energy. Arbitrary local physical properties at β_k is also estimated as

$$\langle\hat{A}\rangle_{\beta_k} = \langle\psi_k|\hat{A}|\psi_k\rangle/N_s. \tag{5.14}$$

In finite-size system, error is caused by the choice of the initial random vector. To estimate the average value and error of the physical properties, we perform some independent calculations by changing $|\psi_0\rangle$.

5.3.1 Details of implementation

Initial vector

For TPQ method, an initial vector is given by using a random generator, i.e. coefficients of all components for the initial vector is given by random numbers. The seed is calculated as

$$123432 + (n_{\rm run} + 1) \times |r_s| + k_{\rm Thread} + N_{\rm Thread} \times k_{\rm Process} \tag{5.15}$$

where r_s is a number given by an input file and $n_{\rm run}$ is a number of runs. r_s and a maximum value of $n_{\rm run}$ are defined by `initial_iv` and `NumAve` in an input file for Standard mode or a ModPara file for Expert mode, respectively. Random numbers are generated by using SIMD-oriented Fast Mersenne Twister (dSFMT) [9]. We can select a type of initial vector from a real number or complex number by using `InitialVecType` in a ModPara file. $k_{\rm Thread}, N_{\rm Thread}, k_{\rm Process}$ indicate the thread ID, the number of threads, the process ID, respectively; the initial vector depends both on `initial_iv` and the number of parallelism.

5.4 Bogoliubov representation

In the spin system, the spin indices in input files of `transfer`, `InterAll`, and correlation functions are specified as those of the Bogoliubov representation. Spin operators are written by using creation/annihilation operators as follows:

$$S_{iz} = \sum_{\sigma=-S}^{S} \sigma c_{i\sigma}^\dagger c_{i\sigma} \tag{5.16}$$

$$S_i^+ = \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1) - \sigma(\sigma+1)} c_{i\sigma+1}^\dagger c_{i\sigma} \tag{5.17}$$

$$S_i^- = \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1) - \sigma(\sigma+1)} c_{i\sigma}^\dagger c_{i\sigma+1} \tag{5.18}$$

6 Acknowledgement

We would like to express our sincere gratitude to Prof. Hidetoshi Nishimori and Mr. Daisuke Tahara. Implementation of the Lanczos algorithm in $\mathcal{H}\Phi$ written in C is based on the pioneering diagonalization package TITPACK written in Fortran by Prof. Nishimori. For developing the user interface of $\mathcal{H}\Phi$, we follow the design concept of the user interface in the program for variational Monte Carlo developed by Mr. Tahara. A part of the user interface in $\mathcal{H}\Phi$ is based on his original codes. We would also like to thank the support from "*Project for advancement of software usability in materials science*" by The Institute for Solid State Physics, The University of Tokyo, for development of $\mathcal{H}\Phi$ ver.0.1, ver.0.2, ver.0.3, ver.1.0, ver.1.1 and ver.1.2.

References

[1] E. Dagotto, Rev. Mod. Phys. **66**, 763–840 (1994).

[2] M. Imada, M. Takahashi, Journal of the Physical Society of Japan **55**, 3354–3361 (1986).

[3] J. Jaklič, P. Prelovšek, Phys. Rev. B **49**, 5065–5068 (1994).

[4] A. Hams, H. De Raedt, Phys. Rev. E **62**, 4365–4377 (2000).

[5] S. Sugiura, A. Shimizu, Phys. Rev. Lett. **108**, 240401 (2012).

[6] Y. Yamaji, Y. Nomura, M. Kurita, R. Arita, M. Imada, Phys. Rev. Lett. **113**, 107201 (2014).

[7] http://www.stat.phys.titech.ac.jp/ nishimori/titpack2_new/index-e.html.

[8] M. Sugihara, K. Murota, Theoretical Numerical Linear Algebra, Iwanami Studies in Advanced Mathematics, Iwanami Shoten, Publishers, 2009.

[9] http://www.math.sci.hiroshima-u.ac.jp/ m-mat/MT/SFMT.