HPhi++ Documentation

Release 3.2.0

HPhi++ team

Nov 27, 2019

Contents

CHAPTER 1

About HPhi++

A numerical solver package for a wide range of quantum lattice models including Hubbard-type itinerant electron hamiltonians, quantum spin models, and Kondo-type hamiltonians for itinerant electrons coupled with quantum spins. The Lanczos algorithm for finding ground states and newly developed Lanczos-based algorithm for finite-temperature properties of these models are implemented for parallel computing. A broad spectrum of users including experimental researchers is cordially welcome.

CHAPTER 2

License

This package is distributed under GNU General Public License version 3 (GPL v3) or later.

We kindly ask you to cite the article

Mitsuaki Kawamura, Kazuyoshi Yoshimi, Takahiro Misawa, Youhei Yamaji, Synge Todo, and Naoki Kawashima

Comp. Phys. Commun. 217 (2017) 180-192.

in publications that include results obtained using this software.

CHAPTER 3

Copyright

CHAPTER 4

Download

You can download software and source codes of HPhi++ from GitHub page or release page.

CHAPTER 5

Contents

5.1 What is HPhi++?

5.1.1 What is HPhi++?

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

For the flexible and quantitative comparison of theoretical and experimental data, the exact diagonalization approach[1] is one of the most reliable numerical tools that requires no approximation or inspiration of genius. For the last few decades, a numerical diagonalization package for quantum spin Hamiltonians, TITPACK, developed by Prof. Hidetoshi Nishimori of Tokyo Institute of Technology, has been widely used in the condensed-matter physics community. Nevertheless, limited computational resources have hindered the ability of non-expert users to apply the package to quantum systems with a large number of electrons or spins.

In contrast, the recent and rapid development of a parallel computing infrastructure has opened 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][3][4][5] allow the finite temperature properties of quantum many-body systems to be calculated at computational costs similar to those of the calculations of ground state properties, which also allows theoretical results for the temperature dependence of, for example, specific heat and magnetic susceptibility, to be compared 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.

HPhi++, a flexible diagonalization package for solving quantum lattice Hamiltonians, has been developed as a descendant of the pioneering package TITPACK. The Lanczos method for calculations of the ground state and a few excited

[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).

states properties, as well as finite temperature calculations based on thermal pure quantum states[5] , are implemented in the HPhi++ package, with an easy-to-use and flexible user interface. By using HPhi++, 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 the SU(2) symmetry of quantum spins, such as Dzyaloshinskii-Moriya and Kitaev interactions, and Kondo lattice models describing itinerant electrons coupled with quantum spins. HPhi++ calculates a variety of physical quantities, such as internal energy at zero temperature or finite temperatures, temperature dependence of specific heat, and charge/spin structure factors. A broad spectrum of users including experimental scientists is cordially welcome.

License

The distribution of the program package and the source codes for HPhi++ follow GNU General Public License version 3 (GPL v3) or later. We hope that you cite the reference, Comp. Phys. Commun. 217 (2017) 180-192 , when you publish the results using HPhi++ (hphi).

Copyright

© 2015- The University of Tokyo. All rights reserved. This software was developed with the support of "*Project for advancement of software usability in materials science*" of The Institute for Solid State Physics, The University of Tokyo.

Contributors

This software was developed by the following contributors.

- ver.3.1 (released on 2018/9/3)
  - Developers
    - Takahiro Misawa
      (The Institute for Solid State 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)
    - Kota Ido
      (Department of Applied Physics, The University of Tokyo)
    - Youhei Yamaji
      (Department of Applied Physics, The University of Tokyo)
    - Synge Todo
      (Department of Physics, The University of Tokyo)
    - Yusuke Konishi
      (Academeia Co., Ltd.)
  - Project coordinator
    - Naoki Kawashima
      (The Institute for Solid State Physics, The University of Tokyo)
- ver.3.0 (released on 2017/12/22)

  - Developers
    - Takahiro Misawa
      (The Institute for Solid State 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)
    - Kota Ido
      (Department of Applied 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.2.0 (released on 2017/4/11)
- ver.1.2 (released on 2016/11/14)
- ver.1.1 (released on 2016/5/13)
- ver.1.0 (released on 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)

5.1.2 Operating environment

HPhi++ was tested on the following platforms

- The supercomputer system-B “sekirei” in ISSP
- Fujitsu FX-10 and K computer

- Linux PC + Intel compiler
- Linux PC + GCC.
- Mac + GCC.

5.2 How to use HPhi++

5.2.1 Prerequisite

HPhi++ requires the following packages:

- C/fortran compiler (Intel, Fujitsu, GNU, etc.)
- BLAS/LAPACK library (Intel MKL, Fujitsu, ATLAS, etc.)
- MPI library (if you do not use MPI, this is not required).

Tip:

E.g. /Settings of Intel compiler
When you use the Intel compiler, you can easily use the scripts attached to the compiler.
In the case of the bash in the 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 the manuals of your compiler/library for more information.

5.2.2 Installation

You can download HPhi++ at the following location.

https://github.com/QLMS/HPhi/releases

You can obtain the HPhi++ directory by typing

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

HPhi++ can be installed by using cmake.

Tip:

Before using cmake for sekirei, you must type
```
source /home/issp/materiapps/tool/env.sh
```

We can compile HPhi++ as:

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

To use ScaLAPACK library for full diagonalization, the cmake option `-DUSE_SCALAPACK=ON` is needed. Here, we set a path to HPhi++ as `$PathTohphi` and to a build directory as `$HOME/build/hphi`. After compilation, `src` folder is constructed below a `$HOME/build/hphi` folder and we obtain an executable `HPhi` in `src/` directory. When no MPI library exists in the system, an executable `HPhi` is automatically compiled without an MPI library.

In the above example, we compile HPhi++ by using a gcc compiler. We can select a compiler by using the following options:

- `sekirei` : ISSP system-B "sekirei"
- `sekirei_acc`: ISSP system-B "sekirei" (for using MAGMA library)
- `fujitsu` : Fujitsu compiler
- `intel` : Intel compiler + Linux PC
- `gcc` : GCC compiler + Linux PC.

An example of compiling HPhi++ by using the Intel compiler is shown as follows:

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

After compilation, `src` folder is created below the `build` folder and an execute HPhi++ in the `src` folder. Please note that we must delete the `build` folder and repeat the above operations when we change the compiler.

5.2.3 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
|     |--sekirei_acc.cmake
|--doc/
|     |--en/
|     |--jp/
|     |--fourier/
|     |--sphinx/
|     |--userguide_en.pdf
|     |--userguide_jp.pdf
|--samples/
|--src/
|     |--*.c
|     |--CMakeLists.txt
|     |--include/*.h
|     |--makefile_src
|     |--StdFace/
```

(continues on next page)

(continued from previous page)

```
|--test/
|--tool/
```

5.2.4 Basic usage

HPhi++ has two modes: Standard mode and Expert mode. Here, the basic flows of calculations of the Standard and expert modes are shown.

Standard mode

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

1. Create a directory for a calculation scenario

   First, you create a working directory for the calculation.

2. Create input files for Standard mode

   In Standard mode, you can choose a model (the Heisenberg model, Hubbard model, etc.) and a lattice (the square lattice, triangular lattice, etc.) from those provided; you can specify some parameters (such as the first/second nearest neighbor hopping integrals and the on-site Coulomb integral) 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 *How to use HPhi*.

3. Run

   Run an executable `HPhi` in the terminal by setting option " `-s`" (or " `--standard`") and the name of the input file written in the 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 to your system manuals. The number of processes depends on the target system of the models. The details of setting the number of processes are shown in *Creating input files for Expert mode* .

4. Watch calculation logs

   Log files are outputted in the "output" folder, which is automatically created in the directory for a calculation scenario. The details of the output files are shown in *Output files* .

5. Results

   If the calculation is completed normally, the result files are outputted in the "output" folder. The details of the output files are shown in *Output files*

Tip:

The number of threads for OpenMP

If you specify the number of OpenMP threads for HPhi++, you should set it as follows (in the case of 16 threads) before running:

```
export OMP_NUM_THREADS=16
```

Expert mode

The calculation procedure for Expert mode is as follows.

1. Create a directory for a calculation scenario

   First, you create a directory and give it the name of a calculation scenario (you can attach an arbitrary name to a directory).

2. Create input files for Expert mode

   For Expert mode, you should create input files for constructing Hamiltonian operators, calculation conditions, and a list file for the filenames of the input files (see the file formats shown in *Input files for Expert mode*).

Note:

A list file can be easily created by using Standard mode.

3. Run

   Run HPhi++ in the terminal by setting option "`-e`" (or "`--expert`") and the 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 *Creating input files for Expert mode*.

4. While running

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

5. Results

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

Creating input files for *Expert* mode

This mode is for creating input files for *Expert* mode. A set of input files created using this mode gives a model provided in *Standard* mode. The usage is shown as follows.

1. Create an input file for *Standard* mode.
2. Setting an option "-sdry" and an input file (in this example, StdFace.def), run HPhi++.

```
$ Path/HPhi -sdry StdFace.def
```

In this case, you should not use MPI parallelization (mpirun, mpiexec, etc.).

3. The following files are created as the input files for *Expert* mode in the current working directory.

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

Setting the process number for MPI/hybrid parallelization

For using MPI/hybrid parallelization, the process number must be set as follows.

1. Standard mode

   - Hubbard/Kondo model

     When `model` in the input file for Standard mode is set as `"Fermion Hubbard"`, `"Kondo Lattice"`, or `"Fermion HubbardGC"`, the process number must be equal to 4^n.

   - Spin model

     When `model` in the input file for Standard mode is set as `"Spin"` or `"SpinGC"`, the process number must be equal to $(2S+1)^n$, where `2S` is set in the input file (the default value is 1).

2. Expert mode

   - Hubbard/Kondo model

     When the model is selected as the Fermion Hubbard model or Kondo model by setting `CalcModel` in a **CalcMod** file, the process number must be equal to 4^n. See *CalcMod file* for details of the `CalcModel` file.

   - Spin model

     When the model is selected as the spin model by setting `CalcModel` in a **CalcMod** file, the process number is fixed by a **LocSpin** file. The process number must be equal to the number calculated by multiplying the state number of the localized spin (`2S` +1) in descending order by the site number. See *LocSpin file* for details of the **LocSpin** file.

     For example, when a **LocSpin** file is given as follows, the 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
```

Printing version ID

By using the `-v` option as follows, you can check which version of HPhi++ you are using.

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

5.3 Tutorial

5.3.1 Quick guide to *Standard* mode

Heisenberg model

This tutorial should be performed in

```
samples/CG/Heisenberg/
```

The input file is provided as follows:

```
samples/CG/Heisenberg/stan.in
```

In this case, we treat the two-dimensional antiferromagnetic Heisenberg model that has a nearest neighbor spin coupling.

$$\hat{\mathcal{H}} = J \sum_{i,j=1}^{4} (\hat{S}_{ij} \cdot \hat{S}_{i+1j} + \hat{S}_{ij} \cdot \hat{S}_{ij+1},)$$

where we use the periodic boundary condition $(S_{15} = S_{51} = S_{11})$.

The input file is as follows:

```
model = "Spin"
method = "CG"
lattice = "square"
W = 4
L = 4
J = 1.0
2Sz = 0
```

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

Log output

Log messages are outputted to the standard output. Log files for the calculation procedure are created in the "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 of the outputted files are shown in *CHECK_Chemi.dat* etc. We execute

```
$ Path/HPhi -s stan.in
```

and obtain the following standard outputs (the compilation mode is MPI parallel/hybrid parallel)

```
      ,ammmmmmmmmmmmmmmb,,      Welcome to the
    ,@@` dm          mb  ===m
  ,@@` d@@@@@@@@@@@@@@@@b Pm,   @@          @@       @@
 d@  d@@@ @@@ @@@@@@ @@@@b ~@a  @@          @@    @@@@@@@@
d@   @@@@ ^^^ @@@@ m m @@@   @, @@          @@  @@@  @@  @@@
@    @@@@_@@@_@@@@mm mm@@@   @| @@mmmmmmmmmm@@ @@    @@    @@
P@    9@@@@@@@@@@@@@@@@@P    @~ @@@@@@@@@@@@@@ @@    @@    @@
 @@      ~~9@@@@@@PPP~      @P  @@          @@  @@@  @@  @@@
  ~@@b      @@@@@@@      ,@@~   @@          @@    @@@@@@@@
    ~@@@m,,@@@@@@@@@  ,m@~`     @@          @@       @@
        ~~9@@@@@@@@@  ~
           9@P~~~9@P            Version 2.0.3


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

  OpenMP threads : 1
  MPI PEs : 1


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

  Open Standard-Mode Inputfile stan.in

  KEYWORD : model                | VALUE : Spin
  KEYWORD : method               | VALUE : CG
  KEYWORD : lattice              | VALUE : square
  KEYWORD : w                    | VALUE : 4
  KEYWORD : l                    | VALUE : 4
  KEYWORD : j                    | VALUE : 1.0
  KEYWORD : 2sz                  | VALUE : 0

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

  @ Lattice Size & Shape

                a = 1.00000     ######  DEFAULT VALUE IS USED  ######
          Wlength = 1.00000     ######  DEFAULT VALUE IS USED  ######
          Llength = 1.00000     ######  DEFAULT VALUE IS USED  ######
               Wx = 1.00000     ######  DEFAULT VALUE IS USED  ######
               Wy = 0.00000     ######  DEFAULT VALUE IS USED  ######
               Lx = 0.00000     ######  DEFAULT VALUE IS USED  ######
               Ly = 1.00000     ######  DEFAULT VALUE IS USED  ######
           phase0 = 0.00000     ######  DEFAULT VALUE IS USED  ######
           phase1 = 0.00000     ######  DEFAULT VALUE IS USED  ######

  @ Super-Lattice setting

                L = 4
                W = 4
           Height = 1           ######  DEFAULT VALUE IS USED  ######
         Number of Cell = 16

  @ Hamiltonian

                h = 0.00000     ######  DEFAULT VALUE IS USED  ######
            Gamma = 0.00000     ######  DEFAULT VALUE IS USED  ######
```

(continues on next page)

(continued from previous page)

```
              2S = 1           ######  DEFAULT VALUE IS USED  ######
               D = 0.00000     ######  DEFAULT VALUE IS USED  ######
             J0x = 1.00000
             J0y = 1.00000
             J0z = 1.00000
             J1x = 1.00000
             J1y = 1.00000
             J1z = 1.00000

  @ Numerical conditions

       LargeValue = 4.50000     ######  DEFAULT VALUE IS USED  ######

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

    locspn.def is written.
    coulombinter.def is written.
    hund.def is written.
    exchange.def is written.
    CDataFileHead = zvo         ######  DEFAULT VALUE IS USED  ######
      Lanczos_max = 2000        ######  DEFAULT VALUE IS USED  ######
       initial_iv = -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  ######
           NOmega = 200         ######  DEFAULT VALUE IS USED  ######
         OmegaMax = 72.00000    ######  DEFAULT VALUE IS USED  ######
         OmegaMin = -72.00000   ######  DEFAULT VALUE IS USED  ######
          OmegaIm = 0.04000     ######  DEFAULT VALUE IS USED  ######
              2Sz = 0
     modpara.def is written.

  @ Spectrum

       SpectrumQW = 0.00000     ######  DEFAULT VALUE IS USED  ######
       SpectrumQL = 0.00000     ######  DEFAULT VALUE IS USED  ######
       SpectrumQH = 0.00000     ######  DEFAULT VALUE IS USED  ######
     SpectrumType = szsz        ######  DEFAULT VALUE IS USED  ######
        pair.def is written.


  @ CalcMod

          Restart = none        ######  DEFAULT VALUE IS USED  ######
   InitialVecType = c           ######  DEFAULT VALUE IS USED  ######
       EigenVecIO = none        ######  DEFAULT VALUE IS USED  ######
         CalcSpec = none        ######  DEFAULT VALUE IS USED  ######
     calcmod.def is written.

      ioutputmode = 1           ######  DEFAULT VALUE IS USED  ######
    greenone.def is written.
    greentwo.def is written.
    namelist.def is written.

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

(continues on next page)

(continued from previous page)

```
  Read File 'namelist.def'.
  Read File 'calcmod.def' for CalcMod.
  Read File 'modpara.def' for ModPara.
  Read File 'locspn.def' for LocSpin.
  Read File 'coulombinter.def' for CoulombInter.
  Read File 'hund.def' for Hund.
  Read File 'exchange.def' for Exchange.
  Read File 'greenone.def' for OneBodyG.
  Read File 'greentwo.def' for TwoBodyG.
  Read File 'pair.def' for PairExcitation.

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

  Read File 'locspn.def'.
  Read File 'coulombinter.def'.
  Read File 'hund.def'.
  Read File 'exchange.def'.
  Read File 'greenone.def'.
  Read File 'greentwo.def'.
  Read File 'pair.def'.

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

  MAX DIMENSION idim_max=12870
  APPROXIMATE REQUIRED MEMORY  max_mem=0.001647 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
```

(continues on next page)

(continued from previous page)

```


######  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.

######  Eigenvalue with LOBPCG  #######

  initial_mode=1 (random): iv = -1 i_max=12870 k_exct =1

    Step   Residual-2-norm     Threshold      Energy
        1     2.44343e+00     1.00000e-07         -5.27456e-01
        2     2.76604e+00     1.87217e-07         -1.87217e+00
        3     2.61923e+00     4.19088e-07         -4.19088e+00
        4     2.57106e+00     5.97098e-07         -5.97098e+00

( snip )

       40     7.39431e-06     1.12285e-06         -1.12285e+01
       41     4.15948e-06     1.12285e-06         -1.12285e+01
       42     2.04898e-06     1.12285e-06         -1.12285e+01
       43     9.92048e-07     1.12285e-06         -1.12285e+01

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


######  End  : Calculate Lanczos EigenVec.  ######

i=    0 Energy=-11.228483 N= 16.000000 Sz=  0.000000 Doublon=  0.000000
```

In the beginning of this run, files describing the details of the considered Hamiltonian (`locspin.def`, `trans.def`, `exchange.def`, `coulombintra.def`, `hund.def`, `namelist.def`, `calcmod.def`, `modpara.def`) and files specifying the elements of the correlation functions that will be calculated(`greenone.def`, `greentwo.def`) are generated.

Outputs for calculation results

Locally Optimal Block Conjugate Gradient (LOBCG) method

When a calculation by the LOBCG 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, the following files are outputted.

```
zvo_energy.dat
zvo_cisajscktalt_eigen_xx.dat  zvo_phys_Nup4_Ndown4.dat
```

where xx is the number of the eigenstate counting from 0.

Lanczos method

When a calculation by the Lanczos method is completed normally, eigenenergies, one-body Green's functions, and two-body Green's functions are calculated and outputted to the files, respectively. In this sample, the 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 the accuracy of the Lanczos vectors is sufficient, one-body and two-body Green's functions are calculated by the eigenvectors obtained by the Lanczos method. When the accuracy of the Lanczos vectors is *not* sufficient, a message "Accuracy of Lanczos vector is not enough" is outputted to the standard output and the one-body and two-body Green's functions are calculated by the eigenvectors obtained by CG method. The details of output files are shown in *energy.dat* , *cisajs.dat* , *cisajscktalt.dat*.

TPQ method

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

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

In Norm_rand%%.dat, basic information such as the inverse of temperature and the norm of the wave function before normalization is outputted with a TPQ step for each number of runs. In SS_rand%%.dat, physical quantities such as the inverse of temperature, energy, and expected value of the square of the 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 *Norm_rand.dat*, *SS_rand.dat*, *cisajs.dat*, *cisajscktalt.dat*.

Full diagonalization method

When `method = "fulldiag"` is selected in an input file, a calculation by the full diagonalization method is started. After the calculation is completed normally, the following files are outputted, where xx is the number of the 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 line.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 the expected values of energy and the doublon are outputted. The details of these files are shown in *Eigenvalue.dat* - *cisajscktalt.dat*.

Other tutorials

There are many tutorials in `samples/Standard/`. For more details, please see `README.md` at each directory.

5.3.2 Quick guide to *Expert* mode

For Expert mode, the 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 the same as in Standard mode. In this section, we demonstrate Expert mode in the directory where the tutorial at the previous section was performed.

File list for input files

In namelist.def, the types of input files and filenames are defined as shown below. By writing the keyword and filenames at each line, the types of files are distinguished. The details of namelist.def are shown in *List file for the input files*.

```
       ModPara  modpara.def
       LocSpin  locspn.def
  CoulombInter  coulombinter.def
          Hund  hund.def
      Exchange  exchange.def
      OneBodyG  greenone.def
      TwoBodyG  greentwo.def
       CalcMod  calcmod.def
PairExcitation  pair.def
   SpectrumVec  zvo_eigenvec_0
```

Files for basic parameters

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

Setting a calculation mode

The calculation mode is set in a CalcMod file (in this sample file, calcmod.def). The contents of the files are as follows.

```
#CalcType = 0:Lanczos, 1:TPQCalc, 2:FullDiag, 3:CG
#CalcModel = 0:Hubbard, 1:Spin, 2:Kondo, 3:HubbardGC, ..
#Restart = 0:None, 1:Save, 2:Restart&Save, 3:Restart
#CalcSpec = 0:None, 1:Normal, 2:No H*Phi, 3:Save, ...
CalcType   3
CalcModel   1
ReStart   0
CalcSpec   0
CalcEigenVec   0
InitialVecType   0
InputEigenVec   0
```

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

Setting parameters for calculation

The parameters for the calculation are set in a ModPara file(in this sample, modpara.def). The contents of this file are as follows.

```
--------------------
Model_Parameters   0
--------------------
HPhi_Cal_Parameters
--------------------
CDataFileHead  zvo
CParaFileHead  zqp
--------------------
Nsite          16
2Sz            0
Lanczos_max    2000
initial_iv     -1
exct           1
LanczosEps     14
LanczosTarget  2
LargeValue     4.500000000000000e+00
NumAve         5
ExpecInterval  20
NOmega         200
OmegaMax       7.200000000000000e+01     4.000000000000000e-02
OmegaMin       -7.200000000000000e+01    4.000000000000000e-02
OmegaOrg       0.0 0.0
```

In this file, we set the parameters for the calculation, such as the site number, the total number of conduction electrons, the total S_z and the number of Lanczos steps. The details of the ModPara file are shown in *ModPara file*.

Setting positions of localized spins

The positions S of the localized spins are defined by a LocSpin file (in this sample, locspn.def). The contents of the files are as follows.

```
================================
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 the spin system, all the sites are automatically treated as localized spins. The details of a LocSpin file are shown in *LocSpin file*.

Files for constructing Hamiltonian

After setting the basic parameters, we create input files for constructing the Hamiltonian. Since the calculations are performed by using the representation of the fermion operators in HPhi++, we must rewrite the spin operator. For example, in the case of $S = 1/2$, we rewrite the equation by using the relation

$$\begin{aligned} S_z^{(i)} &= (c_{i\uparrow}^\dagger c_{i\uparrow} - c_{i\downarrow}^\dagger c_{i\downarrow})/2, \\ S_+^{(i)} &= c_{i\uparrow}^\dagger c_{i\downarrow}, \\ S_-^{(i)} &= c_{i\downarrow}^\dagger c_{i\uparrow}. \end{aligned}$$

Setting transfer integrals

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

$$\mathcal{H}+ = -\sum_{ij\sigma_1\sigma 2} t_{ij\sigma_1\sigma 2} c_{i\sigma_1}^\dagger c_{j\sigma_2}.$$

The contents of the files are as follows.

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

We can use this term when an electric magnetic field is added in the spin system. For example, when a 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 *Trans file*.

Setting general two-body interactions

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

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

The contents of the files are as follows.

```
======================
NInterAll      96
======================
========zInterAll=====
======================
```

(continues on next page)

(continued from previous page)

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

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

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 follows.

```
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 the Hamiltonian. The details of the input formats of two-body interactions are shown in *InterAll file* - *PairLift file*.

Setting output components

In OneBodyG and TwoBodyG files, the indices of one-body and two-body Green's functions are defined, respectively.

Setting indices of one-body Green's functions

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

```
===============================
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 of the input formats of a OneBodyG file are shown in *OneBodyG file*.

Setting indices of two-body Green's functions

In the TwoBodyG file (in this sample, greentwo.def), the indices of $\langle c^{\dagger}_{i\sigma_1} c_{j\sigma_2} c^{\dagger}_{k\sigma_3} c_{l\sigma_4} \rangle$ are defined. The contents of this file are as follows.

```
=============================================
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 of the input formats of the TwoBodyG file are shown in *TwoBodyG file*.

Running

After creating all the input files above, we are 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 HPhi++.

```
$ Path/HPhi -e namelist.def
```

The process after the calculation is the same as that of Standard mode.

5.3.3 Fourier transformation of correlation functions

This package has a utility which performs the Fourier transformation of the correlation function and plots that function. The manual of this utility is located in

```
doc/fourier/ja/_build/html/index.html
doc/fourier/ja/_build/latex/fourier.pdf
doc/fourier/en/_build/html/index.html
doc/fourier/en/_build/latex/fourier.pdf
```

5.4 File specification

5.4.1 Input files for *Standard* mode

An example of an input file for the standard mode is as follows:

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

 lattice = "triangular lattice"
//mu = 1.0
```

(continues on next page)

(continued from previous page)

```
// 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 with a "//"(comment outs) are skipped.
- Upper- and lowercase are not distinguished. Double quotes and blanks are ignored.
- There are three types of parameters.
  1. Parameters that must be specified (if not, HPhi++ will stop with error messages),
  2. Parameters that it is not necessary to specified (if not specified, default values are used),
  3. Parameters that must not be specified (if specified, HPhi++ will stop with error messages).

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

We explain each keyword as follows.

Parameters for the type of calculation

- `model`

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

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

$$\mathcal{H} = -\mu \sum_{i\sigma} c^\dagger_{i\sigma} c_{i\sigma} - \sum_{i \neq j \sigma} t_{ij} c^\dagger_{i\sigma} c_{j\sigma} + \sum_i U n_{i\uparrow} n_{i\downarrow} + \sum_{i \neq j} V_{ij} n_i n_j, \tag{5.1}$$

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

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

  the canonical ensemble in the Kondo lattice model

$$\mathcal{H} = -\mu \sum_{i\sigma} c^\dagger_{i\sigma} c_{i\sigma} - t \sum_{\langle ij \rangle \sigma} c^\dagger_{i\sigma} c_{j\sigma} + \frac{J}{2} \sum_i \left\{ S_i^+ c^\dagger_{i\downarrow} c_{i\uparrow} + S_i^- c^\dagger_{i\uparrow} c_{i\downarrow} + S_{iz}(n_{i\uparrow} - n_{i\downarrow}) \right\}, \tag{5.2}$$

[1] GC=Grand Canonical

the grand canonical ensemble of the Fermion in the Hubbard model [Eqn. (5.1)], the grand canonical ensemble in the Spin model [Eqn. (5.2)], and the grand canonical ensemble in the Kondo lattice model [Eqn. (5.2)], respectively.

When `model="SpinGCCMA"`, by using a more efficient algorithm[2], HPhi++ calculates a system that is 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"`, `"CG"`, `Time Evolution`)

  Description : The calculation type is specified with this parameter; the above expressions above denote the single eigenstate calculation by using the Lanczos method, at the finite-temperature by using the thermally pure quantum state, the full diagonalization method, the multiple eigenstates calculation by using the LOBCG method[3][4] , and the simulation of real-time evolution, respectively.

  The scheme employed for the spectrum calculation is also specified with this parameter. If `"CG"` is chosen, the shifted bi-conjugate gradient method[5] together with the seed-switch technique[6] is employed with the help of the $K\omega$ library[7] .

- `lattice`

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

  Description : The lattice shape is specified with this parameter; the expressions above denote the one-dimensional chain lattice (Fig. 5.1 (a)), the two-dimensional square lattice (Fig. 5.1 (b)), the two-dimensional triangular lattice (Fig. 5.1 (c)), the two-dimensional anisotropic honeycomb lattice (Fig. 5.2), the ladder lattice (Fig. 5.4), and the Kagome Lattice(Fig. 5.3) respectively.

  In `method="SpinGCCMA"`, only `"Chain Lattice"`, `"Honeycomb Lattice"`, `"Ladder"`, and `"Kagome"` are supported. The 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"`

    $W=3, L \geq 2$, $N_{\rm proc} \leq 1 (L=2)$, $N_{\rm proc} \leq 512 (L>2)$.

[2] Y. Yamaji *et. al.*, manuscript in preparation.
[3] A.V.Knyazev, SIAM Journal on Scientific Computing **23**, 517 (2001).
[4] S.Yamada, T.Imamura, M.Machida, The Japan Society for Computational Engineering and Science **2006**, 20060027 (2006).
[5] A.Frommer, Computing **70**, 87{109 (2003).
[6] S.Yamamoto, T. Sogabe, T. Hoshi, S.-L. Zhang, T. Fujiwara, Journal of the Physical Society of Japan **77**, 114713 (2008).
[7] https://github.com/issp-center-dev/Komega.

Parameters for the lattice

Chain [Fig. 5.1 (a)]

- `L`

  Type : Integer

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

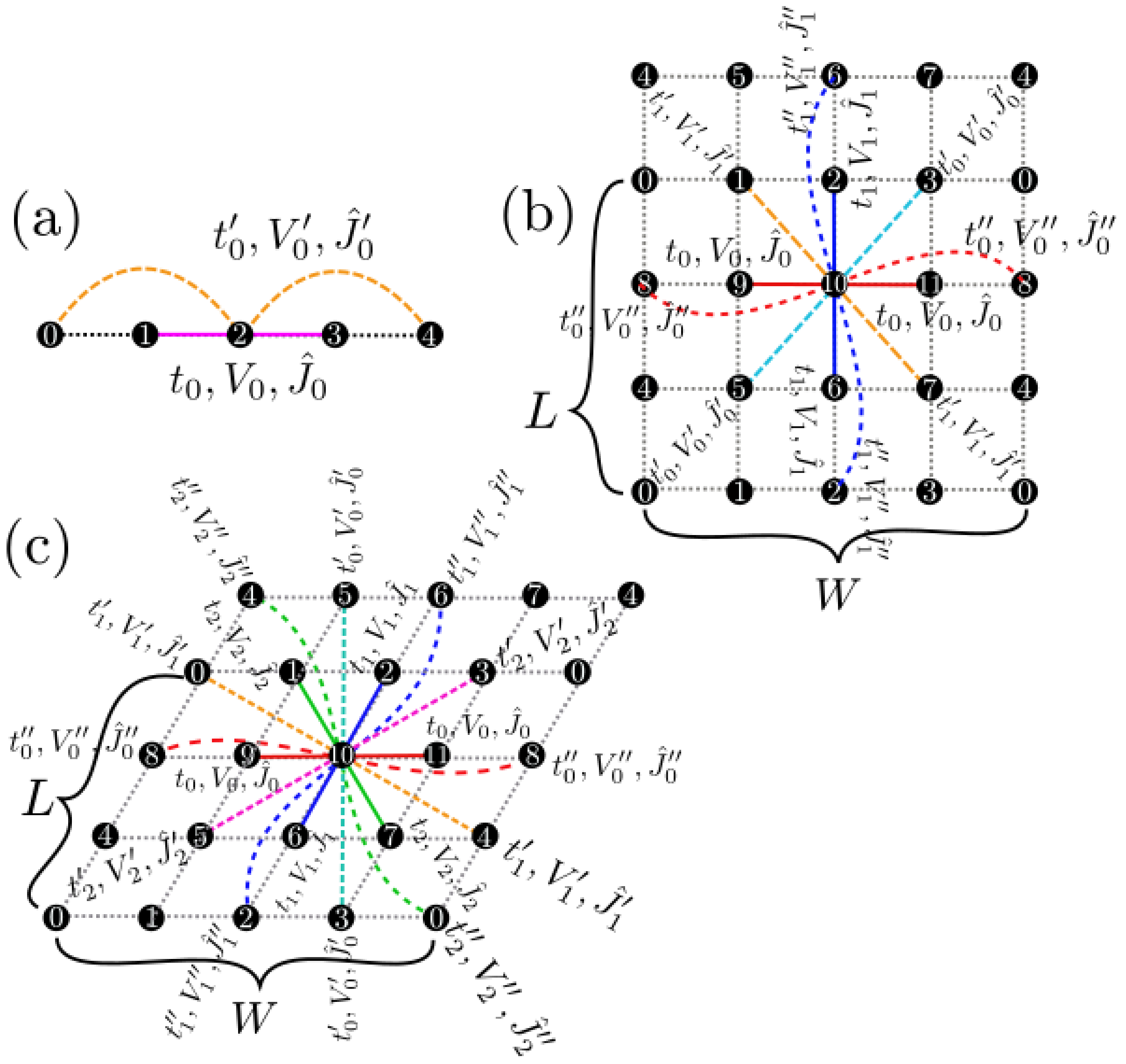

Fig. 5.1: Figure 1: Schematic illustration of (a) one-dimensional chain lattice, (b) two-dimensional square lattice, and (c) two-dimensional triangular lattice. They have t, V, and J as the nearest neighbor hopping, an offsite Coulomb integral, and a spin-coupling constant, respectively (magenta solid lines); they also have t', V', and J' as the next nearest neighbor hopping, offsite Coulomb integral, and spin-coupling constant, respectively (green dashed line).

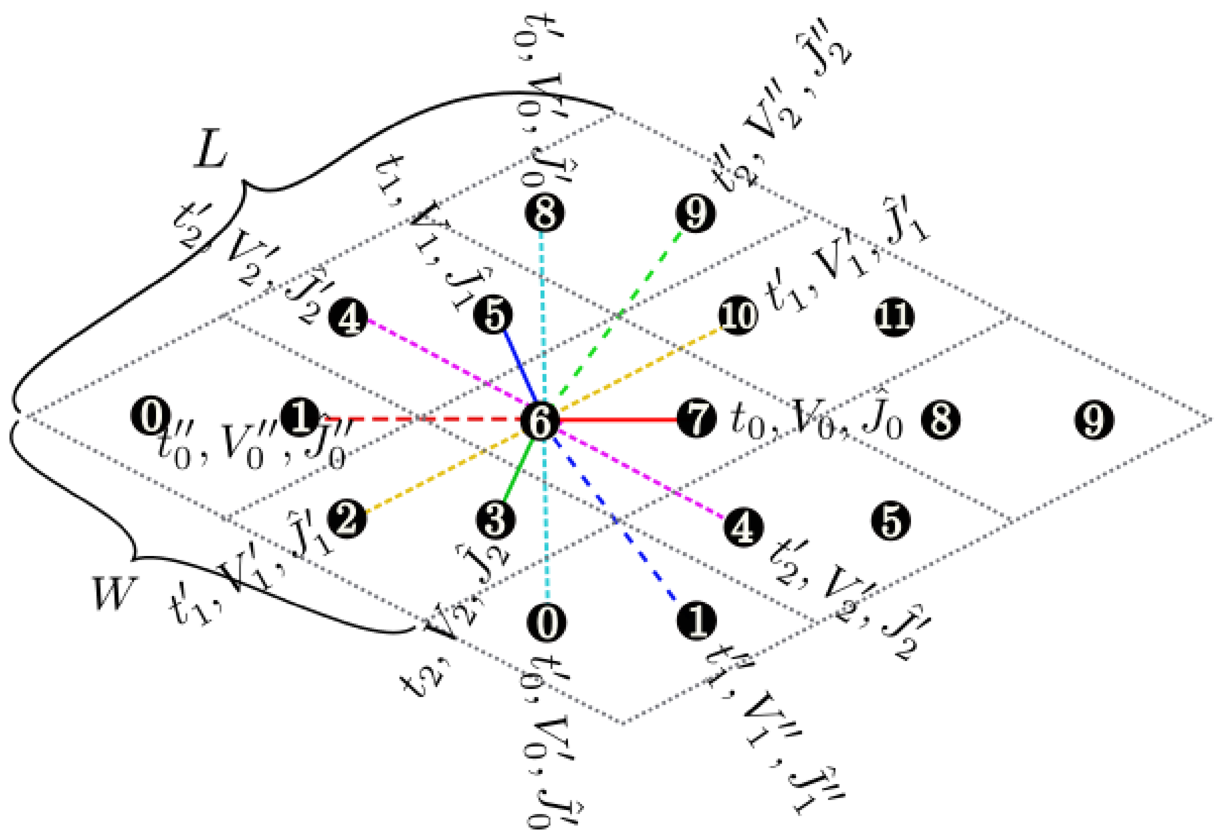

Fig. 5.2: Figure 2: Schematic illustration of the anisotropic honeycomb lattice. The first/second/third nearest neighbor hopping integral, spin coupling, and offsite Coulomb integral depend on the bond direction.

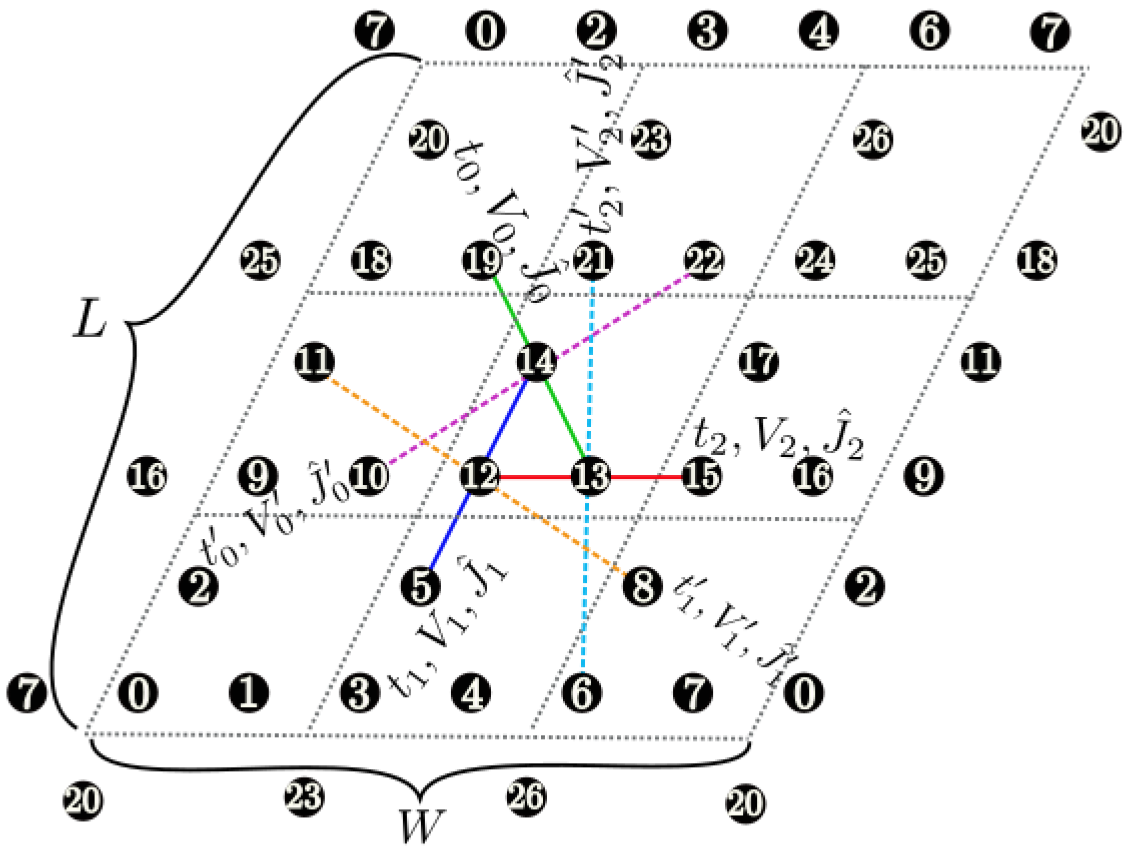

Fig. 5.3: Figure 3: Schematic illustration of the Kagome lattice.

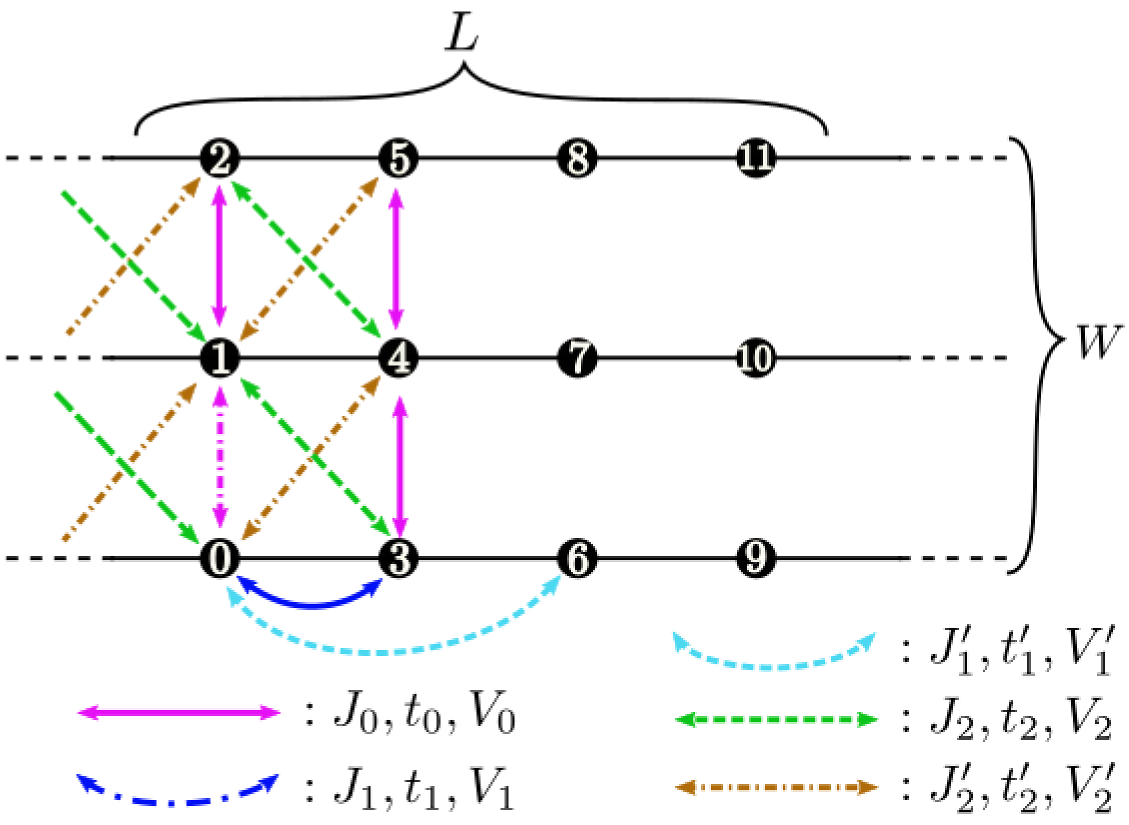

Fig. 5.4: Figure 4: Schematic illustration of the ladder lattice.

Ladder (Fig. 5.4)

- `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.

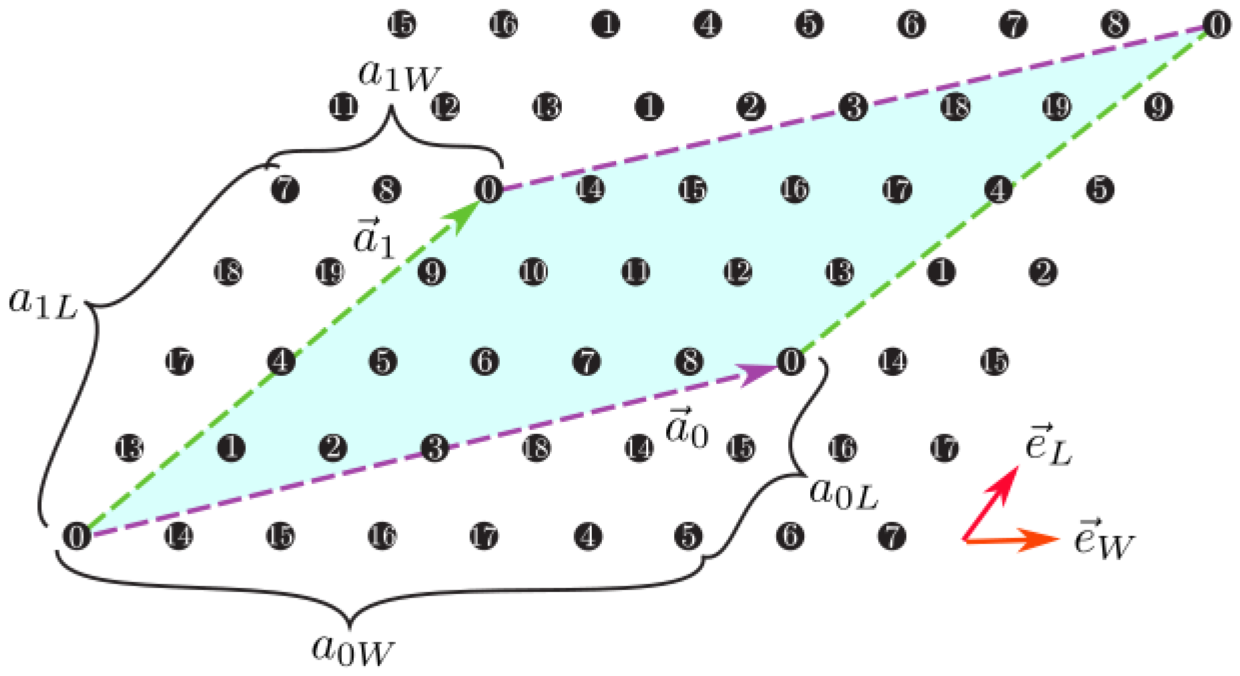

Fig. 5.5: Figure 5: Shape of the numerical cell when $\boldsymbol{a}_0 = (6, 2), \boldsymbol{a}_1 = (2, 4)$ in the triangular lattice. The region surrounded by $\boldsymbol{a}_0$ (magenta dashed arrow) and $\boldsymbol{a}_1$ (green dashed arrow) becomes the cell to be calculated (20 sites).

Tetragonal lattice [Fig. 5.1 (b)], triangular lattice [Fig. 5.1 (c)], honeycomb lattice [Fig. 5.2], Kagome lattice [Fig. 5.3]

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 the original unit cells (dashed black lines in Fig. 5.1 - Fig. 5.3) is specified with this parameter.

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

  Type : Integer

  Description : We can specify two vectors ($\boldsymbol{a}_0, \boldsymbol{a}_1$) that surround the numerical cell (Fig. 5.5). These vectors should be specified in the fractional coordinate.

If we use both these methods, HPhi++ stops. When `model=SpinGCCMA`, 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 Standard mode. This file can be read by `gnuplot` as follows:

```
$ gnuplot lattice.gp
```

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.

Parameters for the Hamiltonian

A default value is 0 unless a specific value is written in the description. Table 5.1 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 : It is available only for the Hubbard and Kondo lattice model. The chemical potential μ (including the site potential) is specified with this parameter.

- `U`

  Type : Real

  Description : It is available only for the 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 : It is available only for the 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, HPhi++ will stop.

- `h`

  Type : Real

  Description : The longitudinal magnetic field is specified with this parameter.

- `Gamma`, `D`

  Type : Real

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

The non-local terms described below should be specified differently according to the lattice structure: For `lattice=Ladder`, the non-local terms are specified differently from those for the `lattice=Chain Lattice`, `Square Lattice`, `Triangular Lattice`, `Honeycomb Lattice`, `Kagome`. Below, the available parameters for each lattice are shown in Table 5.1 .

Table 5.1: Interactions for each model defined in an input file. We can define spin couplings as a matrix format.

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

Non-local terms[for Ladder (Fig. 5.4)]

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

  Type : Complex

  Description : (Hubbard and Kondo lattice model) Hopping integrals in the ladder lattice (see Fig. 5.4) are 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. 5.2) 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. 5.4) are specified with these parameters. 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, HPhi++ will stop. The above rules are also valid for the simplified parameters, `J1`, `J1'`, `J2`, and `J2'`.

Non-local terms [other than Ladder (Fig. 5.1 , Fig. 5.2 , Fig. 5.3)]

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

  Type : Complex

Description : (Hubbard and Kondo lattice model) The nearest neighbor hoppings for each direction (see Fig. 5.1 - Fig. 5.3) are specified with these parameters. 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, HPhi++ will stop.

- `t'`, `t0'`, `t1'`, `t2'`

  Type : Complex

  Description : (Hubbard and Kondo lattice model) The second nearest neighbor hoppings for each direction (see Fig. 5.1 - Fig. 5.3) 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, HPhi++ will stop.

- `t''`, `t0''`, `t1''`, `t2''`

  Type : Complex

  Description : (Hubbard and Kondo lattice model) The third nearest neighbor hoppings for each direction (see Fig. 5.1 - Fig. 5.3) 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, HPhi++ will stop.

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

  Type : Real

  Description : (Hubbard and Kondo lattice model) The nearest neighbor offsite Coulomb integrals V for each direction (see Fig. 5.1 - Fig. 5.3) 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, HPhi++ will stop.

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

  Type : Real

  Description : (Hubbard and Kondo lattice model) The second nearest neighbor-offsite Coulomb integrals V' for each direction (see Fig. 5.1 - Fig. 5.3) are specified with this parameter. 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, HPhi++ will stop.

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

  Type : Real

  Description : (Hubbard and Kondo lattice model) The third nearest neighbor-offsite Coulomb integrals V' for each direction (see Fig. 5.1 - Fig. 5.3) are specified with this parameter. 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, HPhi++ 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) The nearest neighbor exchange couplings for each direction are specified with these 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, HPhi++ 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`-`Jzy`) is specified in addition to its counterparts (`J0x`-`J2zy`), HPhi++ will stop. Below, examples of parameter sets for nearest neighbor exchange couplings are shown.

- If there are no bond-dependent, and 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 there 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 there 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 there 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`.

- `J'x`, `J'y`, `J'z`, `J'xy`, `J'yx`, `J'xz`, `J'zx`, `J'yz`, `J'zy`
- `J0'x`, `J0'y`, `J0'z`, `J0'xy`, `J0'yx`, `J0'xz`, `J0'zx`, `J0'yz`, `J0'zy`
- `J1'x`, `J1'y`, `J1'z`, `J1'xy`, `J1'yx`, `J1'xz`, `J1'zx`, `J1'yz`, `J1'zy`
- `J2'x`, `J2'y`, `J2'z`, `J2'xy`, `J2'yx`, `J2'xz`, `J2'zx`, `J2'yz`, `J2'zy`

  Type : Real

  Description : (Spin model) The second nearest neighbor exchange couplings are specified. However, for `lattice = Honeycomb Lattice` and `lattice = Kagome` with `model=SpinGCCMA`, 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`) are specified, HPhi++ will stop.

- `J''x`, `J''y`, `J''z`, `J''xy`, `J''yx`, `J''xz`, `J''zx`, `J''yz`, `J''zy`
- `J0''x`, `J0''y`, `J0''z`, `J0''xy`, `J0''yx`, `J0''xz`, `J0''zx`, `J0''yz`, `J0''zy`
- `J1''x`, `J1''y`, `J1''z`, `J1''xy`, `J1''yx`, `J1''xz`, `J1''zx`, `J1''yz`, `J1''zy`
- `J2''x`, `J2''y`, `J2''z`, `J2''xy`, `J2''yx`, `J2''xz`, `J2''zx`, `J2''yz`, `J2''zy`

  Type : Real

  Description : (Spin model) The third nearest neighbor exchange couplings are specified. However, for `lattice = Honeycomb Lattice` and `lattice = Kagome` with `model=SpinGCCMA`, the third 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`) are specified, HPhi++ will stop.

- `phase0`, `phase1`

  Type : Double (`0.0` as defaults)

  Description : We can specify the phase for the hopping through the cell boundary with these parameter (unit: degree). These factors for the $\boldsymbol{a}_0$ direction and the $\boldsymbol{a}_1$ direction can be specified independently. For the one-dimensional system, only `phase0` can be used. For example, a fopping from i-th site to j-th site through the cell boundary with the positive direction becomes as

$$\exp(i \times \mathrm{phase0} \times \pi/180) \times t\hat{c}_{j\sigma}^{\dagger}\hat{c}_{i\sigma} + \exp(-i \times \mathrm{phase0} \times \pi/180) \times t^{*}\hat{c}_{i\sigma}^{\dagger}\hat{c}_{j\sigma}$$

Parameters for the numerical condition

- `2S`

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

  Description : The $2S$ at each site in the localized spin system is specified. (E.g. `1` for the $1/2$ system)

- `Restart`

  Type : String (choose from `"None"`, `"Restart_out"`, `"Restart_in"`, `"Restart"`. `"None"` as a default)

  Description : The condition of the restart is specified. `"None"` for omitting file IOs for the restart, `"Restart_out"` for starting calculation from scratch and generating a restart-file after the calculation finishes, `"Restart_in"` for starting calculation with the restart-file generated in the previous run, `"Restart"` for `"Restart_out"` + `"Restart_in"`.

- `anczos_max`

  Type : Positive integer (default value: `2000`)

  Description : The upper limit of the Lanczos/LOBCG/BiCG step and the number of steps for TPQ/Time=evolution are specified with this parameter.

- `initial_iv`

  Type : Integer (default value: `-1`)

  Description : An initial vector is specified with this parameter.

  – Lanczos method

    ∗ For the canonical ensemble and `initial_iv` ≥ 0

      The non-zero components of an initial vector are specified with this parameter.

    ∗ For the grand canonical ensemble or `initial_iv` < 0

      The seed of the random generator is given by this parameter and the random vector is used as the initial vector.

  – TPQ method

    The seed of the random generator is given by this parameter and the random vector is used as the initial vector.

  See *Algorithm* for details of setting an initial vector.

- `exct`

  Type : Positive integer (default value: `1`)

  Description : The number of eigenvectors obtained from the ground energy by the Lanczos method are specified.
  When exct=2, the eigenvector of the first-excited state is obtained. When `method="CG"`, the number of states to be calculated is specified.

  Note: The condition `nvec` $>=$ `exct` must be satisfied.

- `LanczosEps`

  Type : Positive integer (default value: `14`)

Description : The convergence criterion for the Lanczos method is specified with this parameter. If the difference between the old and the new target eigenvalue falls below $10^{-LanczosEps|}$, the Lanczos step will finish. For `method="CG"`, we assume the calculation is converged when the 2-norm of the residual vector becomes smaller than $10^{-\texttt{LanczosEps}/2}$.

- `LanczosTarget`

  Type : Positive integer (default value: `2`)

  Description : The target eigenenergy for the convergence criterion is specified. If it is set to `1`, the target eigenenergy becomes the ground state.

- `LargeValue`

  Type : Double (the default value is provided below)

  Description : (Only for TPQ) l as $l = \hat{\mathcal{H}}/N_s$ is used in the TPQ calculation. Usually, the largest eigenvalue of the Hamiltonian is used as l. Thus, the default value of l is taken as the summation of the absolute values of each coefficient in the Hamiltonian divided by the number of sites.

- `NumAve`

  Type : Positive integer (default value: `5`)

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

- `ExpecInterval`

  Type : Positive integer (default value: `20`)

  Description : (Only for TPQ) The interval of calculating correlation functions in the TPQ iteration is specified.
  Note: A small interval increases the time cost of calculations.

- `OutputMode`

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

  Description : Indices of correlation functions are specified with this keyword. `"none"` indicates correlation functions will not be calculated. When `outputmode="correlation"`, the correlation function supported by the utility `fourier` is computed. For more details, see the document in `doc/fourier/`. If `"full"` is selected, $\langle c_{i\sigma}^\dagger c_{j\sigma'}\rangle$ is computed at all i, j, σ, σ', and $\langle c_{i_1\sigma_1}^\dagger c_{i_2\sigma_2} c_{i_3\sigma_3}^\dagger c_{i_4\sigma_4}\rangle$ is computed at all $i_1, i_2, i_3, i_4, \sigma_1, \sigma_2, \sigma_3, \sigma_4$.

  In a spin system, the indices are specified as those of the Bogoliubov representation (see *Bogoliubov representation*).

- `InitialVecType`

  Type : Character (choose from `"C"`, `"R"`. `"C"` as a default)

  Description : The type of the initial eigenvector is specified. `C` for the complex number, and `R` for the real number.

- `EigenVecIO`

  Type : String (choose from `"None"`, `"Out"`, `"In"`. `"None"` as a default)

  Description : The I/O of the eigenvector is specified. `"None"` for omitting the IO of the eigenvector, `"Out"` for writing the eigenvector to a file, `"In"` for reading the eigenvector from a file and using it in the subsequent calculation (such as the Green's function).

Parameters for the dynamical Green's function

- `CalcSpec`

  Type : String(choose from `"None"`, `"Normal"`, `"NoIteration"`, `"Restart_out"`, `"Restart_in"`, `"Restart"`. `"None"` as default.)

  Description : The condition for the calculation of the dynamical Green's function is specified. `"None"` for omitting the calculation of the dynamical Green's function. `"Normal"` for calculating that function from scratch, `"NoIteration"` for calculating that function with the same iteration in the previous run (In this case, the Hamiltonian-vector product is not performed. Although the numerical cost is very small, the convergence is not guaranteed), `"Restart_out"` for calculating that function from scratch and writing the restart-file at the end, `"Restart_in"` for starting the calculation with the previously written restart-file, `"Restart"` for `"Restart_out"` + `"Restart_in"`.

  The scheme for the spectrum calculation is specified by using the parameter `method`. If `method="CG"` is chosen, the shifted bi-conjugate gradient method[1] together with the seed-switch technique[2] is employed with the help of the $K\omega$ library[3].

- `SpectrumType`

  Type : String (choose from `"SzSz"`, `"S+S-"`, `"Density"`, `"up"`, `"down"`. `"SzSz"` as default.)

  Description : The type of the dynamical Green's function to be computed is specified. `"SzSz"` for $\langle \hat{S}_{zq}\hat{S}_{zq}\rangle$, `"S+S-"` for $\langle \hat{S}^+_q\hat{S}^-_q\rangle$, `"Density"` for $\langle \hat{n}_q\hat{n}_q\rangle$, `"up"` for $\langle \hat{c}^\dagger_{q\uparrow}\hat{c}_{q\uparrow}\rangle$, `"down"` for $\langle \hat{c}^\dagger_{q\downarrow}\hat{c}_{q\downarrow}\rangle$.

- `SpectrumQW`, `SpectrumQL`

  Type : Double (default value: `0.0`)

  Description : The wave number (Fractional coordinate) of the dynamical Green's function is specified. The reciprocal lattice vector is computed from the direct lattice vector shown in Fig. 5.1 , Fig. 5.2 , Fig. 5.4 , Fig. 5.3 .

- `OmegaMin`

  Type : Double (`-LargeValue` times the number of sites as default.)

  Description : The lower limit of the real part of the frequency.

- `OmegaMax`

  Type : Double (`LargeValue` times the number of sites as default.)

  Description : The upper limit of the real part of the frequency.

- `OmegaIm`

  Type : Double (`0.01*LargeValue` as a default.)

  Description : The imaginary part of the frequency.

- `NOmega`

  Type : Positive integer (`200` as a default.)

  Description : The number of frequencies.

[1] A. Frommer, Computing **70**, 87{109 (2003).
[2] S. Yamamoto, T. Sogabe, T. Hoshi, S.-L. Zhang, T. Fujiwara, Journal of the Physical Society of Japan **77**, 114713 (2008).
[3] https://github.com/issp-center-dev/Komega.

Parameters for time-evolution

- `dt`

  Type : Positive Double(Default value : `0.1`)

  Description : The width of time steps.

- `PumpType`

  Type : String (Chosen from `"Quench"`, `"Pulse Laser"`, `"AC Laser"`, and `"DC Laser"`.Default value : `"Quench"`)

  Description : The type of time-dependent Hamiltonian. For `"Quench"`, two body operator $U_{\rm quench} \sum_i n_{i\uparrow} n_{i\downarrow}$ is added. For `"Pulse Laser"`, `"AC Laser"`, and `"DC Laser"`, the hopping term is modulated as $-\sum_{ij\sigma} t_{ij} \exp[-{\bf A}(t) \cdot ({\bf R}_i - {\bf R}_j)/(2\pi)] c_{i\sigma} c_{j\sigma}$, where ${\bf A}(t)$ is the vector potential which is defined as ${\bf A}(t) = {\bf A}_0 \exp[-(t-t_0)^2/(2t_{\rm dump}^2)] \cos[\omega(t-t_0)]$, ${\bf A}(t) = {\bf A}_0 \sin[\omega(t-t_0)]$, and ${\bf A}(t) = {\bf A}_0 t$ for for `"Pulse Laser"`, `"AC Laser"`, and `"DC Laser"`, respectively.

  `potential.dat` file is written for displaying the vector potential and the electrical field at each time step.

- `Uquench`

  Type : Double(Default value : `0.0`)

  Description : $U_{\rm quench}$

- `freq`

  Type : Double(Default value : `0.1`)

  Description : ω

- `tshift`

  Type : Double(Default value : `0.0`)

  Description : t_0

- `tdump`

  Type : Double (Default value : `0.1`)

  Description : $t_{\rm dump}$

- `VecPotW`, `VecPotL`

  Type : Double (Default value : `0.0`)

  Description : The vector potential (fractional coordinate of the reciprocal space) at $t = t_0$. The reciprocal lattice vector is computed from the direct lattice vector shown in Fig. 5.1 , Fig. 5.2 , Fig. 5.4 , Fig. 5.3 .

5.4.2 Input files for *Expert* mode

In this section, the details of the input files for the expert mode are explained. The input files are categorized according to the following four parts.

(1) List: This file is a list of the input file names with the keywords. Each keyword is fixed, but the file names can be determined freely.

(2) Basic parameters:

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

(3) Hamiltonian:

The Hamiltonian for HPhi++ is denoted by the format of the interactions for the electron system. The types of interaction are determined by the following keywords.
Trans: The one-body part, $c_{i\sigma_1}^{\dagger} c_{j\sigma_2}$
InterAll: The general two-body interactions, $c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4}$.

We can set the interactions that are frequently used by the following keywords.
CoulombIntra: On-site Coulomb interactions, $n_{i\uparrow} n_{i\downarrow}$ $(n_{i\sigma} = c_{i\sigma}^{\dagger} c_{i\sigma})$
CoulombInter: Off-site Coulomb interactions, $n_i n_j$ $(n_i = n_{i\uparrow} + n_{i\downarrow})$
Hund: Hund couplings, $n_{i\uparrow} n_{j\uparrow} + n_{i\downarrow} n_{j\downarrow}$

PairHop: Pair hopping couplings, $c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow}$

Exchange: Exchange couplings, $c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{j\downarrow}^{\dagger} c_{i\downarrow}$

Ising: Ising interactions, $S_i\ z S_j\ z$
PairLift: PairLift couplings, $c_{i\uparrow}^{\dagger} c_{i\downarrow} c_{j\uparrow}^{\dagger} c_{j\downarrow}$.

(4) Output:

The target for the output is determined.
OneBodyG : One-body Green's functions, $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$
TwoBodyG : Two-body Green's functions, $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4} \rangle$.

List file for the input files

List file for the input files

This file determines the input filenames, which are correlated with the keywords. The file format is as follows.

```
CalcMod  calcmod.def
ModPara  modpara.def
LocSpin  zlocspn.def
```

(continues on next page)

(continued from previous page)

```
Trans    ztransfer.def
InterAll zinterall.def
OneBodyG zcisajs.def
TwoBodyG    zcisajscktaltdc.def
```

File format

[string01][string02]

Parameters

- [string01]

  Type : String

  Description : Select a word from keywords.

- [string02]

  Type : String

  Description : An input filename that is correlated with the keywords.

Use rules

- After setting keywords at [string01], the half-width state is needed for writing a filename. You can set the filename freely.
- Keywords for input files are shown in Table 5.2
- Essential keywords are “CalcMod”, “ModPara”, and “LocSpin”.
- Keywords can be set in random order.
- If the keywords or filenames are incorrect, the program is terminated.
- When the head of a line is “#”, the line is skipped.

Table 5.2: List of the definition files

Keywords	Details of 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 one-body Green's functions $\langle c_{i\sigma}^{\dagger}c_{j\sigma}\rangle$
TwoBodyG	Output components for two-body Green's functions $\langle c_{i\sigma}^{\dagger}c_{j\sigma}c_{k\tau}^{\dagger}c_{l\tau}\rangle$
SingleExcitation	Operators for generating a single excited state
PairExcitation	Operators for generating a pair excited state
SpectrumVec	An input vector to calculate a restart vector

CalcMod file

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

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

File format

[string01] [int01]

Parameters

- [string01]

  Type : String

  Description : Select a word from keywords.

- [int01]

  Type : Int

  Description : A parameter that is correlated with a keyword.

Use rules

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

Keywords and parameters

The parameters correlated with the keywords are as follows.

- `CalcType`

  Type : Int

  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: LOBCG for the ground state.

- `CalcModel`

  Type : Int

  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 the fermion Hubbard model, you can select the model under the conservation of the 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, set both `NCond` and `2Sz` in the ModPara file.

- `CalcEigenVec`

  Type : Int (default value: 0)

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

- `InitialVecType`

  Type : Int (default value: 0)

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

- `OutputEigenVec`

  Type : Int (default value: 0)

  Description : Select the mode of outputting an eigenvector:
  0: Not output an eigenvector
  1: Output an eigenvector.

- `InputEigenVec`

  Type : Int (default value: 0)

Description : Select the mode of inputting an eigenvector:
0: Not input an eigenvector
1: Input an eigenvector.

- `ReStart`

  Type : Int (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 (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.

- `OutputHam`

  Type : Int (default value: 0)

  Description : Full Diag)Select the mode of outputting Hamiltonian:
  0: not output Hamiltonian.
  1: output Hamiltonian.

- `InputHam`

  Type : Int (default value: 0)

  Description : (Full Diag)Select the mode of inputting Hamiltonian:
  0: not input Hamiltonian.
  1: input Hamiltonian.

- `OutputExcitedVec`

  Type : Int (default value: 0)

Description : Select the mode of outputting an excited vector:
0: Not output an eigenvector
1: Output an eigenvector.

- `Scalapack`

  Type : Int (default value: 0)

  Description : (Full Diag)Select to use ScaLAPACK library for full diagonalization:
  0: not to use ScaLAPACK.
  1: use ScaLAPACK.

- `NGPU`

  Type : Int (default value: 2)

  Description : (Full Diag)Select the number of GPU devices for full diagonalization:
  $\mathcal{H}\Phi$ does not support to use GPU devices at multi-nodes.

ModPara file

This file determines the parameters for calculation. The file format is 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

  Description : Select a word from keywords.

- [string02]

  Type : String (a blank parameter is not allowed)

  Description : Set a header for output files.

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : A parameter that 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 the parameters are needed and the order for the parameters is fixed

Keywords and parameters

In the following, common parameters and parameters for each method are shown.

Common parameters

- `CDataFileHead`

  Type : String (a blank parameter is 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 the characters set by `CDataFileHead`).

- `Nsite`

  Type : Int (positive integer)

  Description : The number of sites.

- `Ncond`

  Type : Int (positive integer)

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

- `2Sz`

  Type : Int (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), set `Ncond`.

- `initial_iv`

  Type : Int

  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 are specified with this parameter.

    * For grand canonical ensemble or `initial_iv` < 0

      The seed of the random generator is given by this parameter and the random vector is used as the initial vector.

  - TPQ method

    The seed of the random generator is given by this parameter and the random vector is used as the initial vector.

  See *Algorithm* for details of setting an initial vector.

- `CalcHS`

  Type : Int (positive integer)

  Description : If CalcHS=1, an efficient algorithm for generating the restricted Hilbert space with the specified quantum number is used (Details of algorithm is shown in http://qlms.github.io/HPhi/develop/tips.pdf [in Japanese]). Default value is 1 and the efficient algorithm is used.

Lanczos method

- `Lanczos_max`

  Type : Int (positive integer)

  Description : The maximum number of Lanczos steps in the calculation. When the convergence within the specified accuracy is satisfied, the calculation is completed before a step reaches `Lanczos_max`. In the case of restart calculation, `Lanczos_max` must be larger than that of the previous calculation.

- `exct`

  Type : Int (positive integer)

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

- `LanczosEps`

  Type : Int (positive integer)

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

- `LanczosTarget`

  Type : Int (positive integer)

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

CG method

- `exct`

  Type : Int (positive integer)

  Description : The number of eigenvectors is specified.

- `Lanczos_max`

  Type : Int (positive integer)

  Description : The maximum number of iteration steps in the calculation. When the convergence within the specified accuracy is satisfied, the calculation is completed before a step reaches Lanczos_max. In the case of restart calculation, `Lanczos_max` must be larger than that of the previous calculation.

- `LanczosEps`

  Type : Int (positive integer)

  Description : For `method="CG"`, the calculation finishes when the 2-norm of the residual vector becomes smaller than $10^{-\texttt{LanczosEps}/2}$.

TPQ method

- `Lanczos_max`

  Type : Int (positive integer)

  Description : The total number of TPQ steps is specified with this parameter. In the case of restart calculation, `Lanczos_max` must be larger than that of the previous calculation.

- `LargeValue`

  Type : Double

  Description : An integer giving l of $l = \hat{\mathcal{H}}/N_s$ used in the TPQ method.

- `NumAve`

  Type : Int

  Description : An integer giving the number of independent runs for the TPQ method.

- `ExpecInterval`

  Type : Int

  Description : An integer giving the interval steps of calculating the correlation functions in the TPQ method. **Note:** A small interval increases the time cost of calculations.

Calculating dynamical Green's functions

- `OmegaOrg`

  Type : Complex

  Description : The center value of the frequency. Specify the real and imaginary parts in that order separated by a space, and if there is no imaginary part, the real part of the frequency is only given.

- `OmegaIm`

  Type : Double

  Description : The imaginary part of the frequency. When `OmegaOrg` is defined in a `modpara` file, `OmegaIm` is added to the imaginary value of `OmegaOrg`.

- `OmegaMin`

  Type : Complex

  Description : The lower limit of the frequency from `OmegaOrg`. Specify the real and imaginary parts in that order separated by a space, and if there is no imaginary part, the real part of the frequency is only given.

- `OmegaMax`

  Type : Complex

  Description : The upper limit of the frequency from `OmegaOrg`. Specify the real and imaginary parts in that order separated by a space, and if there is no imaginary part, a real part of the frequency is only given.

- `NOmega`

  Type : Int

Description : The integer for defining the step size of the frequency $\Delta\omega = ($ `OmegaMax` - `OmegaMin` $)/N_\omega$. The frequency is given by $z_n =$ `OmegaOrg`+`OmegaMin`+$\Delta\omega \times n$.

Real time evolution method

- `Lanczos_max`

  Type : Int (positive integer)

  Description : The total number of real time evolution steps is specified with this parameter. In the case of restart calculation, `Lanczos_max` must be larger than that of the previous calculation.

- `ExpandCoef`

  Type : Int (positive integer)

  Description : An integer giving the expansion order n for real time evolution method;

$$\exp\left(-i\hat{\mathcal{H}}\Delta t\right) = \sum_{i=0}^{N} \frac{1}{n!}\left(-i\hat{\mathcal{H}}\Delta t\right)^n.$$

- `ExpecInterval`

  Type : Int (positive integer)

  Description : An integer giving the interval steps of calculating the correlation functions.
  Note: A small interval increases the time cost of calculations.

- `OutputInterval`

  Type : Int (positive integer)

  Description : An integer giving the interval steps of output the wave function.
  The wave vector is output when `OutputEigenVec=1` in `CalcMod` file.

LocSpin file

This file determines sites with localized spins. The file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of localized spins.

- [int02]

  Type : Int (a blank parameter is not allowed)

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

- [int03]

  Type : Int (a blank parameter is not allowed)

Description : An integer for selecting an electron state whether the electron state is a localized spin or an itinerant electron state:

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].

Trans file

This file determines the values of the transfer integrals $t_{ij\sigma_1\sigma 2}$,

$$\mathcal{H}+ = -\sum_{ij\sigma_1\sigma 2} t_{ij\sigma_1\sigma 2} c_{i\sigma_1}^{\dagger} c_{j\sigma_2}.$$

An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of transfer integrals.

- [int02], [int04]

  Type : Int (a blank parameter is not allowed)

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

- [int03], [int05]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  0: Up-spin
  1: Down-spin.

- [double01]

  Type : Double (a blank parameter is not allowed)

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

- [double02]

  Type : Double (a blank parameter is not allowed)

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

Use rules

- Headers cannot be omitted.
- Since the Hamiltonian must be Hermitian, the relation $t_{ij\sigma_1\sigma_2} = t_{ji\sigma_2\sigma_1}^{\dagger}$ must be satisfied. A program is terminated when this relation is broken.
- A program is terminated when the components of the 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 outside the range of the defined values.

InterAll file

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

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

For spin, the conditions $i = j$ and $k = l$ must be satisfied. An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

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

  Type : Int (a blank parameter is not allowed)

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

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

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  0: Up-spin,
  1: Down-spin.

- [double01]

  Type : Double (a blank parameter is not allowed)

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

- [double02]

  Type : Double (a blank parameter is 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 the Hamiltonian must be Hermitian, the relation $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} = I^{\dagger}_{lkji\sigma_4\sigma_3\sigma_2\sigma_1}$ must be satisfied. A program is terminated when this relation is broken. It is noted that the term of the Hermitian conjugate for $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4}$ should be inputted as $I_{lkji\sigma_4\sigma_3\sigma_2\sigma_1}\ c_{l\sigma_4}^{\dagger} c_{k\sigma_3} c_{j\sigma_2}^{\dagger} c_{i\sigma_1}$.
- A program is terminated when the conditions $i = j$ and $k = l$ are not satisfied for the spin model.
- A program is terminated when the components of the 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 outside the range of the defined values.

CoulombIntra file

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

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

An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

- [int02]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

  Description : A value for U_i.

Use rules

- Headers cannot be omitted.
- A program is terminated when the 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 outside the range of the defined values.

CoulombInter file

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

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

$$Anexampleofthefileformatisasfollows.$$

```
======================
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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

  Description : A value for V_{ij}.

Use rules

- Headers cannot be omitted.
- A program is terminated when the 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] is outside the range of the defined values.

Hund file

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

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

An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of Hund couplings.

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

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

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the 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] is outside the range of the defined values.

PairHop file

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

$$\mathcal{H}+ = \sum_{i,j} J_{ij}^{\rm Pair} (c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow} + h.c.).$$

An example of the file format is as follows.

```
======================
NPairhop 6
======================
========Pairhop ======
======================
   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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of PairHop couplings.

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

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

Use rules

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

Exchange file

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

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

while for the spin system, they are given as

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

We note that $(S_i^+ S_j^- + S_i^- S_j^+)$ **in the 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 the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of Exchange couplings.

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

  Description : A value for J_{ij} Ex.

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the 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] is outside the range of the defined values.

Ising file

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

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

For the spin system, they are given as

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

An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of Ising interactions.

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

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

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the 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] is outside the range of the defined values.

PairLift file

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

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

An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of PairLift couplings.

- [int02], [int03]

  Type : Int (a blank parameter is not allowed)

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

- [double01]

  Type : Double (a blank parameter is not allowed)

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

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the 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] is outside the range of the defined values.

OneBodyG file

This file determines the target components of the one-body Green's function $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$. An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

- [int02], [int04]

  Type : Int (a blank parameter is not allowed)

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

- [int03], [int05]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  For Hubbard or Kondo system, the index can be selected from
  0: Up-spin,
  1: Down-spin.
  For Spin system, the index can be selected from
  $0, 1, \cdots, 2S+1$ (corresponding to -$S-0.5, -S+0.5, \cdots S+0.5$).

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the 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 outside the range of the defined values.

TwoBodyG file

This file determines the target components of the 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 the spin system, the conditions $i = j$ and $k = l$ must be satisfied. An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

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

  Type : Int (a blank parameter is not allowed)

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

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

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  For Hubbard or Kondo system, the index can be selected from
  0: Up-spin,
  1: Down-spin.
  For Spin system, the index can be selected from
  $0, 1, \cdots, 2S+1$ (corresponding to -$S-0.5, -S+0.5, \cdots S+0.5$).

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the two-body Green's functions are double counted.
- A program is terminated when the conditions $i=j$ and $k=l$ are not satisfied for the 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 outside the range of the defined values.

SingleExcitation file

The operators to generate the single excited state $c_{i\sigma_1}(c_{i\sigma_1}^{\dagger})$ are defined. An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

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

- [int02]

  Type : Int (a blank parameter is not allowed)

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

- [int03]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  For Hubbard or Kondo system, the index can be selected from
  0: Up-spin,
  1: Down-spin.
  For Spin system, the index can be selected from
  $0, 1, \cdots, 2S + 1$ (corresponding to -$S - 0.5, -S + 0.5, \cdots S + 0.5$).

- [int04]

  Type : Int (a blank parameter is not allowed)

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

- [double01], [double02]

  Type : Double (a blank parameter is not allowed)

  Description : [double01] gives the real part of $c_{i\sigma_1}(c_{i\sigma_1}^\dagger)$, while [double02] gives the imaginary part of $c_{i\sigma_1}(c_{i\sigma_1}^\dagger)$.

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the single excitation operators are double counted.
- A program is terminated when the conditions $i = j$ 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 outside the range of the defined values.

PairExcitation file

The operators to generate the pair excited state $c_{i\sigma_1}c_{j\sigma_2}^{\dagger}(c_{i\sigma_1}^{\dagger}c_{j\sigma_2})$ are defined. The type of pair excitation operators ($c_{i\sigma_1}c_{j\sigma_2}^{\dagger}$ or $c_{i\sigma_1}^{\dagger}c_{j\sigma_2}$) must be same in the input file. In the S_z conserved system, σ_1 must be equal to σ_2. An example of the file format is 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 (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of pair excitation operators.

- [int02], [int04]

  Type : Int (a blank parameter is not allowed)

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

- [int03], [int05]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  For Hubbard or Kondo system, the index can be selected from
  0: Up-spin,
  1: Down-spin.

For Spin system, the index can be selected from
$0, 1, \cdots, 2S+1$ (corresponding to -$S-0.5, -S+0.5, \cdots S+0.5$).

- [int06]

  Type : Int (a blank parameter is 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 (a blank parameter is not allowed)

  Description : [double01] gives the real part of $c_{i\sigma_1} c_{j\sigma_2}^{\dagger}(c_{i\sigma_1}^{\dagger} c_{j\sigma_2})$, while [double02] gives the imaginary part of $c_{i\sigma_1} c_{j\sigma_2}^{\dagger}(c_{i\sigma_1}^{\dagger} c_{j\sigma_2})$.

Use rules

- Headers cannot be omitted.
- A program is terminated when the components of the pair excitation operators are double counted.
- A program is terminated when the conditions $i = j$ and $k = l$ are not satisfied for the 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 outside the range of the defined values.

SpectrumVec File

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

File name

- ##_rank_$$.dat

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

File format

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

Parameters

- [int01]

  Type : Int

  Description : The total number of the targets of the Hilbert spaces.

- [int02]

  Type : Int

  Description : The number of Lanczos or TPQ steps.

- [double01], [double02]

  Type : double

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

OneBodyTE File

This file determines the values of the transfer integrals $t_{ij\sigma_1\sigma_2}(t)$ at each time t,

$$\mathcal{H}(t) += -\sum_{ij\sigma_1\sigma_2} t_{ij\sigma_1\sigma_2}(t) c_{i\sigma_1}^{\dagger} c_{j\sigma_2}.$$

An example of the file format is as follows.

```
===================================
AllTimeStep          100
===================================
===== OneBody Time Evolution =====
===================================
    0.0   4
    0   0   1   0       1.0     0.0
    1   0   0   0       1.0     0.0
    0   1   1   1       1.0     0.0
    1   1   0   1       1.0     0.0
    0.2   4
   (continue...)
```

File format

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

From line 6, time t and total number of transfer integrals $N(t)$ are first defined and next the transfer integrals $t_{ij\sigma_1\sigma_2}(t)$ are defined.

- Line m: [double01] [int02]
- Lines (m+1) - (m+1+[int02]): [int03] [int04] [int05] [int06] [double02] [double03]

Parameters

- [string01]

  Type : String (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total time steps defined in this file.

- [double01]

  Type : Double (a blank parameter is not allowed)

  Description : Time t.

- [int02]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of transfer integrals at time t.

- [int03], [int05]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a site index ($0 <=$ [int03], [int05] $<$ `Nsite`).

- [int04], [int06]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  0: Up-spin
  1: Down-spin.

- [double02], [double03]

  Type : Double (a blank parameter is not allowed)

  Description : A value for real and imaginary part of $t_{ij\sigma_1\sigma_2}(t)$ at time t is defined [double02] and [double03], respectively.

Use rules

- Headers cannot be omitted.
- A program is terminated when `LanczosStep` defined in `ModPara` is greater than [int02].
- A program is terminated when [int03]-[int06] are outside the range of the defined values.

TwoBodyTE File

This file determines the values of the two-body interactions $I_{i\sigma_1 j\sigma_2 k\sigma_3 l\sigma_4}(t) c_{i\sigma_1}^\dagger c_{j\sigma_2} c_{k\sigma_3}^\dagger c_{l\sigma_4}$ at each time t,

$$\mathcal{H}(t) + = \sum_{ijkl} \sum_{\sigma_1\sigma_2\sigma_3\sigma_4} I_{i\sigma_1 j\sigma_2 k\sigma_3 l\sigma_4}(t) c_{i\sigma_1}^\dagger c_{j\sigma_2} c_{k\sigma_3}^\dagger c_{l\sigma_4}.$$

An example of the file format is as follows.

```
===================================
AllTimeStep          100
===================================
===== TwoBody Time Evolution =====
===================================
    0.0   3
    0  1  0  1  0  0  0  0       1.0    0.0
    1  0  1  0  0  0  0  0       1.0    0.0
    1  0  1  0  2  0  2  0       1.0    0.0
    0.2   3
   (continue...)
```

File format

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

From line 6, time t and total number of transfer integrals $N(t)$ are first defined and next the two-body interactions $I_{i\sigma_1 j\sigma_2 k\sigma_3 l\sigma_4}(t)$ are defined.

- Line m: [double01] [int02]
- Lines (m+1) - (m+1+[int02]): [int03] [int04] [int05] [int06] [int07] [int08] [int09] [int10] [double02] [double03]

Parameters

- [string01]

  Type : String (a blank parameter is not allowed)

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

- [int01]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total time steps defined in this file.

- [double01]

  Type : Double (a blank parameter is not allowed)

  Description : Time t.

- [int02]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving the total number of two-body interactions at time t.

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

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a site index ($0 <=$ [int03],[int05],[int07],[int09] $<$ `Nsite`).

- [int04], [int06], [int08], [int10]

  Type : Int (a blank parameter is not allowed)

  Description : An integer giving a spin index:
  0: Up-spin
  1: Down-spin.

- [double02], [double03]

  Type : Double (a blank parameter is not allowed)

  Description : A value for real and imaginary part of $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}(t)$ at time t is defined [double02] and [double03], respectively.

Use rules

- Headers cannot be omitted.
- A program is terminated when `LanczosStep` defined in `ModPara` is greater than [int02].
- A program is terminated when [int03]-[int10] are outside the range of the defined values.

5.4.3 Output files

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

CHECK_Chemi.dat

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

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

An example of the file format is 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

  Description : The counted number of inputting terms.

- [int02]

  Type : Int

  Description : An integer for showing the spin index of $\mu_{i\sigma}$:
  0: Up-spin,
  1: Down-spin.

- [int03]

  Type : Int

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

- [double01]

  Type : Double

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

CHECK_InterAll.dat

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

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

An example of the file format is 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

  Description : The counted number of inputting terms.

- [int02], [int04]

  Type : Int

  Description : An integer for showing the 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

  Description : An integer for showing the 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

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

CHECK_CoulombIntra.dat

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

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

An example of the file format is 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

  Description : The counted number of inputting terms.

- [int02]

  Type : Int

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

- [double01]

  Type : Double

  Description : A value for U_i.

CHECK_Hund.dat

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

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

An example of the file format is 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

  Description : The counted number of inputting terms.

- [int02], [int03]

  Type : Int

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

- [double01]

  Type : Double

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

CHECK_INTER_U.dat

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

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

An example of the file format is 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

  Description : The counted number of inputting terms.

- [int02], [int03]

  Type : Int

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

- [double01]

  Type : Double

  Description : A value for V_{ij}.

CHECK_Memory.dat

This file shows the size of the memory used in the calculation. An example of the file format is 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

  Description : An integer to show the total numbers of the Hilbert space under a calculation.
- [double01]

  Type : Double

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

WarningOnTransfer.dat

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

  Description : The integer of the site number where the transfer integrals are double counted.

- [int03], [int04]

  Type : Int

  Description : The integer of the spin index of a transfer integral:
  0: Up-spin
  1: Down-spin.

CalcTimer.dat

The name of the calculation process, the process number, and the calculation process time are outputted in order at each line in the CalcTime.dat file. An example of the file format for the TPQ method is 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
================================================
```

TimeKeeper.dat

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

sz_TimeKeeper.dat

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

Time_CG_EigenVector.dat

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

energy.dat

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

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

For `method="CG"`

```
State 0
  Energy  -7.1043675920
  Doublon  0.4164356536
  Sz  0.0000000000

State 1
:
```

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

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

- [double02]

  Type : Double

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

- [double03]

  Type : Double

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

Lanczos_Step.dat

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

```
LanczosStep  E[1] E[2] E[3] E[4] Target:E[3] E_Max/Nsite
stp = 2 1.2149586211 14.6560471044 xxxxxxxxxx xxxxxxxxx 0.0000000000 xxxxxxxxx
stp=4 -5.6626980051 3.1523174817 12.4860778911 21.2322666770 12.4860778911 2.
↪6540333346
stp=6 -8.5113374325 -2.3219709559 4.3459108959 11.5079386600 4.3459108959 3.0307814358
stp=8 -9.5061025854 -5.2757708534 -0.1734100333 5.2236216333 -0.1734100333 3.
↪2049774861
stp=10 -9.7541889139 -6.6054773893 -2.9493235242 1.2364826532 -2.9493235242 3.
↪2686702753
...
stp=84 -10.4543987874 -9.8960493865 -9.7550111859 -9.7407358084 -9.7550111859 3.
↪3731105157
stp=86 -10.4543987874 -9.8960493865 -9.7550111859 -9.7407358084 -9.7550111859 3.
↪3731105157
stp=88 -10.4543987874 -9.8960493865 -9.7550111859 -9.7407358084 -9.7550111859 3.
↪3731105157
```

For `method="CG"`

```
  Step   Residual-2-norm     Threshold      Energy
      1      6.79819e+00     8.19743e-07    7.86586e+00     8.19743e+00     8.02804e+00
      2      7.47402e+00     3.69905e-07    3.35827e+00     3.63546e+00     3.69905e+00
      3      5.30943e+00     2.44472e-07   -2.44472e+00    -2.23296e+00    -1.95487e+00
      4      4.52737e+00     5.10297e-07   -5.10297e+00    -4.92390e+00    -4.58682e+00
      5      3.66168e+00     7.14105e-07   -7.14105e+00    -6.91226e+00    -6.44532e+00
      6      3.12717e+00     8.27201e-07   -8.27201e+00    -7.93262e+00    -7.44680e+00
...
    152      1.05602e-06     1.04544e-06   -1.04544e+01    -9.89605e+00    -9.89605e+00
    153      1.07401e-06     1.04544e-06   -1.04544e+01    -9.89605e+00    -9.89605e+00
    154      9.45018e-07     1.04544e-06   -1.04544e+01    -9.89605e+00    -9.89605e+00
```

File name

- ##_Lanczos_Step.dat

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

File format

- For‘‘method=”Lanczos”‘‘

  stp= [int01] [double01] [double02] [double03] [double04] [double-a] [double-b]

- For `method="CG"`

  [int01] [double-c] [double-d] [double01] [double02] . . .

Parameters

- [int01]

  Type : Int

  Description : The iteration number of the Lanczos and LOBCG method.

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

  Type : Double

  Description : Eigenvalues computed with the Lanczos or LOBCG method (ascending order). Four and `exct` eigenvalues are printed for the Lanczos and LOBCG method, respectively (in the above case, `exct=3`). While the degenerate eigenstates are printed as a single state in the Lanczos method, they are printed separately in LOBCG method. In the above case, we can find there is a degeneracy in the first excited state.

- [double-a]

  Type : Double

  Description : (Only for the Lanczos method) The eigenvalue used for the convergence check. It was specified by `LanczosTarget` (in the above case, `LanczosTarget=3`).

- [double-b]

  Type : Double

  Description : (Only for the Lanczos method) The maximum eigenvalue divided by the number of sites. It is the lower limit of `LargeValue` in TPQ method.

- [double-c]

  Type : Double

  Description : (Only for LOBCG method) The maximum of the 2-norm of each residual vector. It is used for the convergence check.

- [double-d]

  Type : Double

  Description : (Only for LOBCG method) The convergence threshold. This is obtained by the value specified with `LanczosEps` times the absolute value of the energy of the ground state.

Time_TPQ_Step.dat

(For the TPQ method) This file is outputted to show the time for starting the calculation of the TPQ method at each seed and step. In the restart calculation, the values are added to the previous file. An example is 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

  Description : The seed number in the calculation of the TPQ method.

- [int02]

  Type : Int

  Description : The step number in the calculation of the TPQ method.

- [string01]

  Type : String

  Description : The time for starting the calculation of the TPQ method at each seed and step.

Norm_rand.dat

(For the TPQ method) This file is outputted to show the calculation process information for the TPQ method. In the restart calculation, the values are added to the previous file. An example of the file format is 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 the number of runs in the calculation of the TPQ method.

File format

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

Parameters

- [double01]

  Type : Double

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

- [double02]

  Type : Double

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

- [double03]

  Type : Double

  Description : The 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

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

SS_rand.dat

(For the TPQ method) This file is outputted to show the calculation results for the TPQ method. In the restart calculation, the values are added to the previous file. An example of the file format is 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 the number of runs in the calculation of the TPQ method.

File format

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

Parameters

- [double01]

  Type : Double

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

- [double02]

  Type : Double

  Description : The expected value of the energy $\langle\mathcal{H}\rangle$.

- [double03]

  Type : Double

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

- [double03]

  Type : Double

  Description : The expected value of the doublon, $\sum_i\langle n_{i\uparrow}n_{i\downarrow}\rangle$.

- [double05]

  Type : Double

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

- [int01]

  Type : Int

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

Flct_rand.dat

(For the TPQ method) This file is outputted to show the calculation results of the fluctuation of the particle number, doublon, and S_z for the TPQ method. In the restart calculation, the values are added to the previous file. An example of the file format is 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 the number of runs in the calculation of the TPQ method.

File format

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

Parameters

- [double01]

  Type : Double

  Description : Inverse temperature $1/k_{\rm B}T$.
- [double02]

  Type : Double

  Description : A total particle number $\sum_i \langle \hat{n}_i \rangle$.
- [double03]

  Type : Double

  Description : The expected value of the square of the particle number $\langle (\sum_i \hat{n}_i)^2 \rangle$.
- [double04]

  Type : Double

  Description : The expected value of doublon $\frac{1}{N_s} \sum_i \langle n_{i\uparrow} n_{i\downarrow} \rangle$ (N_s is the total number of sites).
- [double05]

  Type : Double

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

- [double06]

  Type : Double

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

- [double07]

  Type : Double

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

- [int01]

  Type : Int

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

Time_TE_Step.dat

(For the real time evolution method) This file is outputted to show the time for starting the calculation of the real time evolution method at each step. In the restart calculation, the values are added to the previous file. An example is as follows.

```
step 1:TE begins: Wed Jul 13 07:59:20 2017
step 2:TE begins: Wed Jul 13 07:59:20 2017
step 3:TE begins: Wed Jul 13 07:59:20 2017
...
step 1997:TE begins: Wed Jul 13 07:59:32 2017
step 1998:TE begins: Wed Jul 13 07:59:32 2017
step 1999:TE begins: Wed Jul 13 07:59:32 2017
```

File name

- ##_TE_Step.dat

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

File format

- stp [int01]: TE begins: [string01]

Parameters

- [int01]

  Type : Int

  Description : The step number in the calculation.

- [string01]

  Type : String

  Description : The time for starting the calculation at each step.

Norm.dat

(For the real time evolution method) This file is outputted to show the calculation process information for the real time evolution method. In the restart calculation, the values are added to the previous file. An example of the file format is as follows.

```
 # time, global_norm, global_1st_norm, step_i
0.0000000000000000 0.9999999999999994 0.0000000000000000 0
0.0100000000000000 1.0000233421898765 0.0000000000000000 1
0.0200000000000000 1.0000211100654208 0.0000000000000000 2
0.0300000000000000 1.0000182214014706 0.0000000000000000 3
0.0400000000000000 1.0000148460317946 0.0000000000000000 4
0.0500000000000000 1.0000111372562455 0.0000000000000000 5
0.0600000000000000 1.0000072252313270 0.0000000000000000 6
0.0700000000000000 1.0000032174168609 0.0000000000000000 7
0.0800000000000000 0.9999992048382456 0.0000000000000000 8
0.0900000000000000 0.9999952720176869 0.0000000000000000 9
0.1000000000000000 0.9999915078951970 0.0000000000000000 10
```

File name

- Norm.dat

File format

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

Parameters

- [double01]

  Type : Double

  Description : Time t

- [double02]

  Type : Double

  Description : The norm of a wave function at t.

- [int01]

  Type : Int

  Description : Time step.

SS.dat

(For the real time evolution method) This file is outputted to show the calculation results. In the restart calculation, the values are added to the previous file. An example of the file format is as follows.

```
 # time, energy, phys_var, phys_doublon, phys_num, step_i
0.0000000000000000  -6.0412438187293001 38.8635272290786489 ...
0.0100000000000000  -5.9310482979751606 37.9593669819686923 ...
0.0200000000000000  -5.8287182777288828 37.1390062697724801 ...
0.0300000000000000  -5.7384311863736031 36.4282319427381651 ...
0.0400000000000000  -5.6636677030535481 35.8466140292489897 ...
0.0500000000000000  -5.6070659264425551 35.4081795274108799 ...
0.0600000000000000  -5.5703150294725914 35.1222606981659666 ...
0.0700000000000000  -5.5540895348193438 34.9942503380419154 ...
0.0800000000000000  -5.5580244678717312 35.0260574979670665 ...
0.0900000000000000  -5.5807312981978212 35.2161499389042660 ...
0.1000000000000000  -5.6198544688797947 35.5591788356216298 ...
...
```

File name

- SS.dat

File format

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

Parameters

- [double01]

  Type : Double

  Description : Time t.

- [double02]

  Type : Double

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

- [double03]

  Type : Double

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

- [double04]

  Type : Double

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

- [double05]

  Type : Double

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

- [int01]

  Type : Int

  Description : Time step.

Flct.dat

(For the real time evolution method) This file is outputted to show the calculation results of the fluctuation of the particle number, doublon, and S_z. In the restart calculation, the values are added to the previous file. An example of the file format is as follows.

```
 # time, N, N^2, D, D^2, Sz, Sz^2, step_i
0.0000000000000000 7.9999999999999991 63.9999999999999929 ...
0.0100000000000000 8.0000000000000604 64.0000000000004832 ...
0.0200000000000000 8.0000000000000018 64.0000000000000142 ...
0.0300000000000000 8.0000000000001013 64.0000000000008100 ...
0.0400000000000000 7.9999999999999183 63.9999999999993463 ...
0.0500000000000000 7.9999999999999520 63.9999999999996163 ...
0.0600000000000000 7.9999999999999627 63.9999999999997016 ...
0.0700000000000000 8.0000000000000835 64.0000000000006679 ...
0.0800000000000000 8.0000000000000924 64.0000000000007390 ...
0.0900000000000000 7.9999999999999600 63.9999999999996803 ...
0.1000000000000000 7.9999999999999067 63.9999999999992539 ...
...
```

File name

- Flct.dat

File format

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

Parameters

- [double01]

  Type : Double

  Description : Time t

- [double02]

  Type : Double

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

- [double03]

  Type : Double

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

- [double04]

  Type : Double

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

- [double05]

  Type : Double

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

- [double06]

  Type : Double

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

- [double07]

  Type : Double

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

- [int01]

  Type : Int

  Description : Time step.

Eigenvalue.dat

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

  Description : The index of eigenvalues. The index 0 is for the energy of the ground state and the indexes are labeled in descending order for energies.

- [double01]

  Type : Double

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

phys.dat

(For the FullDiag method) This file is outputted to show the physical values calculated by the FullDiag method. The data are outputted in descending order for energies. An example of the file format is 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

  Description : The energy $\langle\mathcal{H}\rangle$.

- [double02]

  Type : Double

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

- [double03]

  Type : Double

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

- [double04]

  Type : Double

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

- [double05]

  Type : Double

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

ham.dat

(For the FullDiag method) When `OutputHam=1` in the `CalcMod` file, the Hamiltonian calculated by HPhi++ is outputted by the MatrixMarket format. The recalculation by using this file can be down when `InputHam=1` in the `CalcMod` file. An example of the file format is as follows.

```
%%%%MatrixMarket matrix coordinate complex hermitian
28 28 56
1 1 1.000000 0.000000
2 1 0.500000 0.000000
3 2 0.500000 0.000000
4 3 0.500000 0.000000
5 4 0.500000 0.000000
6 5 0.500000 0.000000
7 6 0.500000 0.000000
7 7 1.000000 0.000000
    ...
```

File name

- ##_ham.dat

## indicates [string02] in a ModPara file.

File format

- Line 1: Header
- Line 2: [int01] [int02] [int03]
- Lines3-:[int04] [int05] [double01] [double02]

Parameters

- [int01]

  Type : Int

  Description : Number of rows of Hamiltonian.

- [int02]

  Type : Int

  Description : Number of columns of Hamiltonian.

- [int03]

  Type : Int

  Description : Number of nonzero elements of Hamiltonian.

- [double01], [double02]

  Type : Double

Description : The value of Hamiltonian; [double01] and [double02] represent real and imaginary part of the Hamiltonian, respectively.

cisajs.dat

This file is the outputted files for one-body Green's function $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$. The target components are set in the input file with the keyword "OneBodyG". An example of the file format is 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, LOBCG method: ##_cisajs_eigen&&.dat.
- Real time evolution method: ## cisajs step%%.dat

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

File format

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

Parameters

- [int01], [int03]

  Type : Int

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

- [int02], [int04]

  Type : Int

  Description : The 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

Description : The value of $\langle c^{\dagger}_{i\sigma_1} c_{j\sigma_2} \rangle$.

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

cisajscktalt.dat

This file is the outputted files for the two-body Green's function $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4} \rangle$. The target components are set in the input file with the keyword "TwoBodyG". An example of the file format is 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, LOBCG method: ##_cisajscktalt_eigen&&.dat
- Real time evolution method: ## cisajscktalt step%%.dat

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

File format

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

Parameters

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

  Type : Int

  Description : The 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

  Description : The 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

Description : The value of $\langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4} \rangle$.

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

eigenvec.dat

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

File name

- ##_eigenvec_&&_rank_$$.dat

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

File format

This file is written through the following source code (a little different fron the actual HPhi++ source).

```
fp = fopen("zvo_eigenvec_0_rank_0.dat", "wb");
fwrite(&number_of_interations, sizeof(int), 1,fp);
fwrite(&local_size, sizeof(unsigned long int),1,fp);
fwrite(&eigen_vector[0], sizeof(complex double),local_size+1, fp);
fclose(fp);
```

where `number_of_interations` is the number of iterations, `local_size` is the size of eigenvector (if MPI is used, it differs from the dimension of the Hilbert space), `eigen_vector` is the (complex) eigenvector.

Note: The fist component of `eigen_vector` (`eigen_vector[0]`) is not used for calculation.

tmpvec.dat

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

File name

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

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

File format

This file is written through the following source code (a little different from the actual HPhi++ source).

Lanczos

```
fp = fopen("zvo_tmpvec_rank_0.dat", "wb");
fwrite(&number_of_interations, sizeof(int), 1,fp);
fwrite(&local_size, sizeof(unsigned long int),1,fp);
fwrite(&last_vector[0], sizeof(complex double),local_size+1, fp);
fwrite(&second_last_vector[0], sizeof(complex double),local_size+1, fp);
fclose(fp);
```

TPQ and LOBPCG

```
fp = fopen("zvo_tmpvec_set_0_rank_0.dat", "wb");
fwrite(&number_of_interations, sizeof(int), 1,fp);
fwrite(&local_size, sizeof(unsigned long int),1,fp);
fwrite(&last_vector[0], sizeof(complex double),local_size+1, fp);
fclose(fp);
```

where `number_of_interations` is the number of iterations, `local_size` is the size of eigenvector (if MPI is used, it differs from the dimension of the Hilbert space), `last_vector` is the vector at the last iteration and `second_last_vector` is the vector at the second last iteration.

Note: The fist component of `last_vector` and `second_last_vector` (`last_vector[0]` and `second_last_vector[0]`) are not used for calculation.

DynamicalGreen.dat

This file is the outputted file for calculating the dynamical Green's function. An example of the file format is as follows.

File name

- ##_DynamicalGreen.dat

## indicates [string02] in a ModPara file.

File format

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

Parameters

- [double01], [double02]

  Type : Double

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

- [double03], [double04]

  Type : Double

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

recalcvec.dat

This file is the outputted file for two vectors to recalculate the dynamical Green's function by the Lanczos method. The file format is of the binary type. An example of the file format is as follows.

File name

- ##_recalcvec_rank_$$.dat

## indicates [string02] in a ModPara file and $$ is the number of rank.

File format

- Line 1: [int01]
- Line 2: [int02]
- Lines 3 - 3+ [int02]: [double01] [double02]
- Lines 4+ [int02] - 4+2 $\times$ [int02]: [double03] [double04].

Parameters

- [int01]

  Type : Int

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

- [int02]

  Type : Long Int

  Description : The total number of targets of the Hilbert spaces.

- [double01], [double02]

  Type : Double

  Description : The value of the vector $\boldsymbol{v}_{k+1}$ for recalculating dynamical Green's functions by the Lanczos method.
  [double01] and [double02] are a real part and an imaginary part of $\boldsymbol{v}_{k+1}$, respectively. The fist component is not used for calculation.

- [double03], [double04]

  Type : Double

  Description : The value of the vector $\boldsymbol{v}_k$ for recalculating dynamical Green's functions by the Lanczos method.
  [double03] and [double04] are a real part and an imaginary part of $\boldsymbol{v}_k$, respectively. The fist component is not used for calculation.

TMcomponents.dat

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

File name

- ##_TMcomponents.dat

## indicates [string02] in a ModPara file and $$ is the number of rank.

File format

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

Parameters

- [int01]

  Type : Int

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

- [double01]

  Type : Double

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

- [double02], [double03]

  Type : Double

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

eigenvec.dat

When OutputExcitedVec=1 in a CalcMod file, the excited vector calculated by using the input vector and excited operators defined in parr.def is outputted. An example of the file format is as follows.

```
4096
0.0009135367, -0.0004231751
-0.0009712430, 0.0293999045
-0.0294644178, 0.0210086435
0.0214247977, -0.0107587147
0.0272643022, -0.0404634256
0.0034322694, -0.0184446640
0.0019911098, 0.0004403706
0.0114685735, -0.0114381935
0.0092888239, 0.0088235535
     ...
```

File name

- ##_excitedvec_&&_rank_$$.dat

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

File format

- Line 1:[int01]
- Lines2-:[double01] [double02]

Parameters

- [int01]

  Type : Int

  Description : Dimenssions of the excited vector.

- [double01], [double02]

  Type : Double

  Description : The value of the excited vector; [double01] and [double02] represent real and imaginary part of the excited vector, respectively.

5.4.4 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.

5.5 Algorithm

5.5.1 Lanczos method

Details of Lanczos method

Some parts of this section are based on the manual of TITPACK[1] and the textbook published by M. Sugihara and K. Murota[2] (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 the Lanczos method by using only two vectors, the dimensions of which are the dimensions of the total Hilbert space[3] , the Lanczos method is frequently used for the diagonalization of the large matrices. As explained in 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 the Krylov subspace. The initial vector is represented by the superposition of the eigenvectors $\boldsymbol{e}_i$ (the corresponding eigenvalues are E_i) of $\hat{\mathcal{H}}$ as

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

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

$$\hat{\mathcal{H}}^n\boldsymbol{x}_0 = E_0^n\Big[a_0\boldsymbol{e}_0 + \sum_{i\neq 0}\left(\frac{E_i}{E_0}\right)^n a_i\boldsymbol{e}_i\Big].$$

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 appropriate transformation for the 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 define an initial vector and 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:

$$\begin{aligned}\alpha_k &= (\hat{\mathcal{H}}\boldsymbol{v}_k,\boldsymbol{v}_k),\\ \boldsymbol{w} &= \hat{\mathcal{H}}\boldsymbol{v}_k - \beta_k\boldsymbol{v}_{k-1} - \alpha_k\boldsymbol{v}_k,\\ \beta_{k+1} &= |\boldsymbol{w}|,\\ \boldsymbol{v}_{k+1} &= \frac{\boldsymbol{v}_k}{|\boldsymbol{v}_k|}.\end{aligned}$$

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

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

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

Here, V_n is a matrix whose column vectors are $\boldsymbol{v}_i(i = 0,1,\dots,n-1)$. T_n is a 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

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

[2] M. Sugihara, K. Murota, Theoretical Numerical Linear Algebra, Iwanami Stud-ies in Advanced Mathematics, Iwanami Shoten, Publishers, 2009.

[3] In HPhi++, to reduce the numerical cost, we use some additional vectors; a vector for accumulating the real-space diagonal elements of the Hamiltonian and a vector for specifying the given S_z space and given particle space. The dimension of these vectors is that of the Hilbert space.

eigenvalues of T_n for sufficiently large n. (We note that $V^\dagger V = I$, I is an identity matrix). The original eigenvectors of $\hat{\mathcal{H}}$ are obtained by $\boldsymbol{e}_i = V\tilde{\boldsymbol{e}}_i$, where $\tilde{\boldsymbol{e}}_i$ denotes 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) $\times$ (the number 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[4] . From this procedure, we obtain the eigenvectors from V.

In the Lanczos method, within a few hundred or thousand Lanczos iterations, we obtain accurate eigenvalues near the maximum and minimum eigenvalues. The necessary number of iterations is sufficiently small as compared to the dimensions of the total Hilbert space.

We note that it is shown that the errors of the maximum and minimum eigenvalues become exponentially small as a function of Lanczos iterations (for details, see Ref.[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 linear simultaneous equations such as

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

By solving this equation 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 four additional vectors are necessary to perform the CG method. For a large system size, it may be impossible to allocate memory to the additional vectors.

Details of implementation

Initial vector

For the 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. The type of initial vector can be selected as 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 the Hilbert space is given by

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

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

- For a grand canonical ensemble or `initial_iv` < 0 The initial vector is given by using a random generator, i.e., the coefficients of all the components for the initial vector are given by random numbers. The seed is calculated as

  $$123432 + |r_s|,$$

  where r_s is the number given by an input file and $n_{\rm run}$ is the number of runs. The 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)[5] .

[4] Upper bound of the dimensions of $\tilde{\boldsymbol{e}_i}$ is # of Lanczos iterations.

[5] http://www.math.sci.hiroshima-u.ac.jp/~m-mat/MT/SFMT/index.html

Convergence condition

In HPhi++, we use `dsyev` (routine of LAPACK) for diagonalization of T_n. We use the energy of the first excited state of T_n as the criterion of convergence. In the standard setting, after five Lanczos steps, we diagonalize T_n every two Lanczos steps. 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 the Lanczos method is not sufficient, 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 frequently accelerates the convergence.

5.5.2 Full Diagonalization method

Overview

We generate the matrix of $\hat{\mathcal{H}}$ by using the real space configuration $|\psi_j\rangle (j = 1 \cdots d_{\rm H}$, where $d_{\rm H}$ is the dimension of the Hilbert space): $\mathcal{H}_{ij} = \langle\psi_i|\hat{\mathcal{H}}|\psi_j\rangle$. 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 a LAPACK routine, such as `dsyev` or `zheev`. We also calculate and output the expectation values $A_i \equiv \langle\Phi_i|\hat{A}|\Phi_i\rangle$. These values are used for the finite-temperature calculations.

Finite-temperature calculations

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

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

The calculation should be performed by using the own postscripts.

5.5.3 Finite-temperature calculations by the TPQ method

Sugiura and Shimizu showed that it is possible to calculate the finite-temperature properties from a few wavefunctions (in the thermodynamic limit, only one wave function is necessary)[1] . The wavefunction is called the thermal pure quantum (TPQ) state. Because the TPQ state can be generated by operating the Hamiltonian to the random initial wavefunction, we directly use the routine Lanczos method for the TPQ calculations. Here, we explain how to construct the micro canonical TPQ (mTPQ) state, which offers the simplest method for calculating finite-temperature properties.

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

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

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

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

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

where u_k is the internal energy. The arbitrary local physical properties at β_k are also estimated as

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

In a 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 $|\psi_0\rangle$.

Details of implementation

Initial vector

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

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

where r_s is the number given by an input file and $n_{\rm run}$ is the number of runs. r_s and the 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)[2] . We can select the type of initial vector as a real number or complex number by using `InitialVecType` in a ModPara file. $k_{\rm Thread}, N_{\rm Thread}, and k_{\rm Process}$ indicate the thread ID, number of threads, process ID, respectively; the initial vector depends both on `initial_iv` and the number of parallelisms.

5.5.4 Dynamical Green's function

Using HPhi++, we can calculate a dynamical Green's function

$$I(z) = \langle \Phi' | \frac{1}{\mathcal{H} - z\hat{I}} | \Phi' \rangle,$$

where $|\Phi'\rangle = \hat{O}|\Phi_0\rangle$ is an excited state and $\hat{O}$ is an excitation operator defined as a single excitation operator

$$\sum_{i,\sigma_1} A_{i\sigma_1} c_{i\sigma_1} (c_{i\sigma_1}^\dagger)$$

or a pair excitation operator

$$\sum_{i,j,\sigma_1,\sigma_2} A_{i\sigma_1 j\sigma_2} c_{i\sigma_1} c_{j\sigma_2}^\dagger (c_{i\sigma_1}^\dagger c_{j\sigma_2}).$$

For example, the dynamical spin susceptibilities can be calculated by defining $\hat{O}$ as

$$\hat{O} = \hat{S}(\mathbf{k}) = \sum_j \hat{S}_j^z e^{i\mathbf{k}\cdot\mathbf{r_j}} = \sum_j \frac{1}{2}(c_{j\uparrow}^\dagger c_{j\uparrow} - c_{j\downarrow}^\dagger c_{j\downarrow}) e^{i\mathbf{k}\cdot\mathbf{r_j}}.$$

There are two modes implemented in $\mathcal{H}\Phi$. One is the continued fraction expansion method by using Lanczos method [1] and the other is the shifted Krylov method[2] . See the reference for the details of each algorithm.

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

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

[2] S.Yamamoto, T. Sogabe, T. Hoshi, S.-L. Zhang, T. Fujiwara, Journal of the Physical Society of Japan **77**, 114713 (2008).

5.5.5 Real time evolution method

In HPhi++, real time evolution calculation is done by using the following relation

$$|\Phi(t_n)\rangle = \exp^{-i\mathcal{H}\Delta t_n} |\Phi(t_{n-1})\rangle,$$

where $|\Phi(t_0)\rangle$ is an initial wave function and $t_n = \sum_{j=1}^{n} \Delta t_j$. In calculation, we approximate $\exp^{-i\mathcal{H}\Delta t_n}$ as

$$\exp^{-i\mathcal{H}\Delta t_n} = \sum_{l=0}^{m} \frac{1}{l!}(-i\mathcal{H}\Delta t_n)^l.$$

Here, the cut-off integer m can be set by *ExpandCoef* in *ModPara*. We can judge whether the expansion order is enough or not by checking the norm conservation $\langle\Phi(t_n)|\Phi(t_n)\rangle = 1$ and energy conservation $\langle\Phi(t_n)|\hat{\mathcal{H}}|\Phi(t_n)\rangle = E$.

5.5.6 Bogoliubov representation

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

$$\begin{aligned}
S_{iz} &= \sum_{\sigma=-S}^{S} \sigma c_{i\sigma}^\dagger c_{i\sigma} \\
S_i^+ &= \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1) - \sigma(\sigma+1)} c_{i\sigma+1}^\dagger c_{i\sigma} \\
S_i^- &= \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1) - \sigma(\sigma+1)} c_{i\sigma}^\dagger c_{i\sigma+1}
\end{aligned}$$

5.6 Calculation of physical quantities

In this chapter, we explain post procedures to calculate physical quantities. All scripts used in this chapter are included in `tool` folder.

5.6.1 Calculation of specific heat

5.6.2 Calculation of spectrum function

5.6.3 Calculation of magnetization curve

.1 HPhi/mVMC Fourier-Transformation utility

.1.1 Overview

This document is the manual for the utility to perform the Fourier transformation of the correlation function in the site representation generated by mVMC or HPhi++.

Prerequisite

The prerequisite of this utility is the same as that of mVMC or HPhi++.

Supported quantities

This utility supports the Fourier transformation of the following quantities:

One-body correlations

$$\langle \hat{c}^{\dagger}_{\mathbf{k}\alpha\uparrow} \hat{c}_{\mathbf{k}\beta\uparrow} \rangle \equiv \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle \hat{c}^{\dagger}_{\mathbf{0}\alpha\uparrow} \hat{c}_{\mathbf{R}\beta\uparrow} \rangle \tag{3}$$

$$\langle \hat{c}^{\dagger}_{\mathbf{k}\alpha\downarrow} \hat{c}_{\mathbf{k}\beta\downarrow} \rangle \equiv \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle \hat{c}^{\dagger}_{\mathbf{0}\alpha\downarrow} \hat{c}_{\mathbf{R}\beta\downarrow} \rangle \tag{4}$$

Density-density correlation

$$\langle \hat{\rho}_{\mathbf{k}\alpha} \hat{\rho}_{\mathbf{k}\beta} \rangle \equiv \frac{1}{N_{\mathbf{R}}} \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle (\hat{\rho}_{\mathbf{0}\alpha} - \langle \hat{\rho}_{\mathbf{0}\alpha} \rangle)(\hat{\rho}_{\mathbf{R}\beta} - \langle \hat{\rho}_{\mathbf{R}\beta} \rangle) \rangle \tag{5}$$

Spin-Spin correlations

$$\langle \hat{S}^z_{\mathbf{k}\alpha} \hat{S}^z_{\mathbf{k}\beta} \rangle \equiv \frac{1}{N_{\mathbf{R}}} \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle \hat{S}^z_{\mathbf{0}\alpha} \hat{S}^z_{\mathbf{R}\beta} \rangle \tag{6}$$

$$\langle \hat{S}^+_{\mathbf{k}\alpha} \hat{S}^-_{\mathbf{k}\beta} \rangle \equiv \frac{1}{N_{\mathbf{R}}} \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle \hat{S}^+_{\mathbf{0}\alpha} \hat{S}^-_{\mathbf{R}\beta} \rangle \tag{7}$$

$$\langle \hat{\mathbf{S}}_{\mathbf{k}\alpha} \cdot \hat{\mathbf{S}}_{\mathbf{k}\beta} \rangle \equiv \frac{1}{N_{\mathbf{R}}} \sum_{\mathbf{R}}^{N_{\mathbf{R}}} e^{-i\mathbf{k}\cdot\mathbf{R}} \langle \hat{\mathbf{S}}_{\mathbf{0}\alpha} \cdot \hat{\mathbf{S}}_{\mathbf{R}\beta} \rangle \tag{8}$$

.1.2 Tutorial

In this tutorial, we explain through a sample calculation of the 8-site Hubbard model on the square lattice.

Run HPhi/vmc.out

- For HPhi++

  We calculate the ground state and the correlation function with the following input file

```
a0w = 2
a0l = 2
a1w = -2
a1l = 2
model="Hubbard"
method="CG"
lattice="square"
t=1.0
U=8.0
nelec = 8
2Sz=0
```

```
$ HPhi -s input
```

- For mVMC

  First, we optimize the trial wavefunction with the following input

```
a0w = 2
a0l = 2
a1w = -2
a1l = 2
model="Hubbard"
lattice="square"
t=1.0
U=8.0
nelec = 8
2Sz=0
```

```
$ vmc.out -s input
```

We add the following line to the input file to compute the correlation function.

```
NVMCCalMode = 1
```

Compute the correlation function.

```
$ vmc.out -s input output/zqp_opt.dat
```

Then the one- and two-body correlation function are written to files in the `output/` directory.

Related files

- StdFace.def (See the manuals for mVMC/HPhi++)
- zqp_opt.dat (See the manual for mVMC)
- greenone.def (*Specify the index of correlation function to be computed*)
- greentwo.def (*Specify the index of correlation function to be computed*)

Fourier transformation of correlation functions

Perform the Fourier transformation of the correlation function by using the utility `greenr2k`.

```
$ echo "4 20
G 0 0 0
X 0.5 0 0
M 0.5 0.5 0
G 0 0 0
16 16 1" >> geometry.dat
$ greenr2k namelist.def geometry.dat
```

Then the Fourier-transformed correlation functions are written to a file in `output/`.

Related files

- output/zvo_cisajs_001.dat (*Results of correlation function in the site representation*)
- output/zvo_cisajs.dat (*Results of correlation function in the site representation*)
- output/zvo_cisajscktalt_001.dat (*Results of correlation function in the site representation*)
- output/zvo_cisajscktalt.dat (*Results of correlation function in the site representation*)
- geometry.dat (*Geometry*)
- output/zvo_corr.dat (*Correlation functions on the k path*)

Display correlation functions

Plot the correlation function in the *k* space by using gnuplot.

```
load "kpath.gp"
plot "output/zvo_corr_eigen0.dat" u 1:12 w l
```

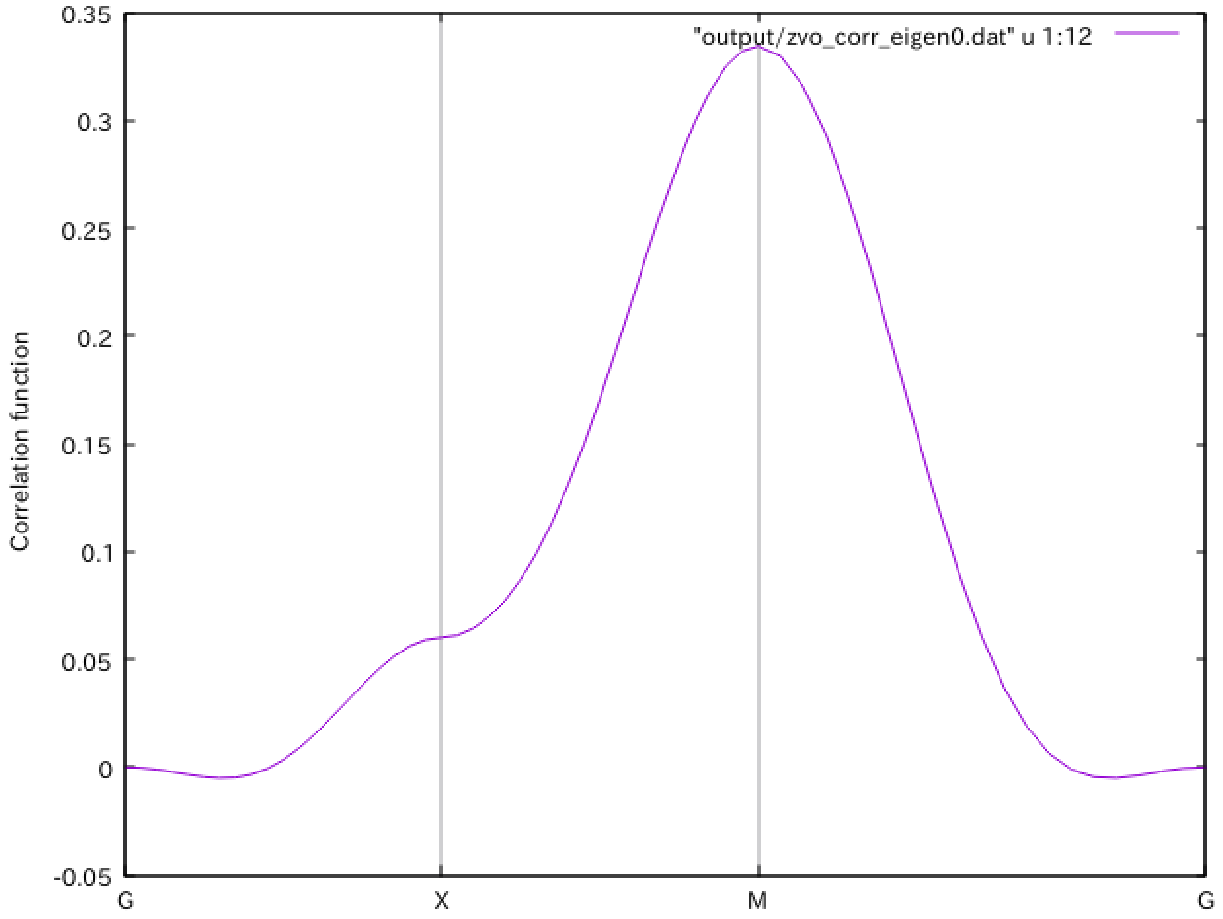

Fig. 6: Figure 6: The spin-spin correlation $\langle \mathbf{S_k} \cdot \mathbf{S_k} \rangle$ (Column 12).

Related files

- kpath.gp (*gnuplot script*)
- output/zvo_corr.dat (*Correlation functions on the k path*)

.1.3 File format

Geometry

The file name in the *Tutorial* is `geometry.dat`. When we use Standard mode of mVMC/HPhi++, the information of the cell and geometry is generated automatically.

```
1.000000000000000e+00     0.000000000000000e+00     0.000000000000000e+00  (1)
0.000000000000000e+00     1.000000000000000e+00     0.000000000000000e+00  (1)
0.000000000000000e+00     0.000000000000000e+00     1.000000000000000e+00  (1)
0.000000000000000e+00     0.000000000000000e+00     0.000000000000000e+00  (2)
2  2  0         (3)
-2  2  0        (3)
0  0  1         (3)
0  0  0  0      (4)
-1  1  0  0     (4)
0  1  0  0      (4)
1  1  0  0      (4)
-1  2  0  0     (4)
0  2  0  0      (4)
1  2  0  0      (4)
0  3  0  0      (4)
4  20           (5)
G  0  0  0      (6)
X  0.5  0  0    (6)
M  0.5  0.5  0  (6)
G  0  0  0      (6)
16  16  1       (7)
```

1. The unit lattice vectors. Arbitrary unit (Generated by Standard mode).
2. The phase for the one-body term across boundaries of the simulation cell (degree unit, Generated by Standard mode).
3. Three integer vector specifying the shape of the simulation cell. They are the same as the input parameters `a0W`, `a0L`, `a0H`, `a1W...` in Standard mode (Generated by standard mode).
4. The index of the lattice vector and the orbital each site (Generated by standard mode).
5. The number of *k* node (high-symmetry point) and the number of *k* along high symmetry line.
6. Fractional coordinate of *k* nodes.
7. The *k* grid to plot the isosurface of the momentum distribution function.

One- and Two-body correlation function in the site representation

Specify the index of correlation function to be computed

Specify the index of correlation functions computed with mVMC/HPhi++. When we use the standard mode, this file is generated automatically. The general description is written in the manuals for mVMC/HPhi++. The file names in the *Tutorial* are `greenone.def` (one body) and `greentwo.def` (two body).

For calculating correlation functions in *Supported quantities*, indices must be specified as follows:

- $\langle \hat{c}^{\dagger}_{\mathbf{k}\alpha\uparrow} \hat{c}_{\mathbf{k}\beta\uparrow} \rangle$

  $\langle \hat{c}^{\dagger}_{\mathbf{0}\alpha\uparrow} \hat{c}_{\mathbf{R}\beta\uparrow} \rangle$ with $\mathbf{R}$ ranging on the all unit cell, and (α, β) ranging on the all orbitals in the unit cell.

- $\langle \hat{c}^{\dagger}_{\mathbf{k}\alpha\downarrow} \hat{c}_{\mathbf{k}\beta\downarrow} \rangle$

  $\langle \hat{c}^{\dagger}_{\mathbf{0}\alpha\downarrow} \hat{c}_{\mathbf{R}\beta\downarrow} \rangle$ with $\mathbf{R}$ ranging on the all unit cell, and (α, β) ranging on the all orbitals in the unit cell.

- $\langle \hat{\rho}_{\mathbf{k}\alpha} \hat{\rho}_{\mathbf{k}\beta} \rangle$ and $\langle \hat{S}^z_{\mathbf{k}\alpha} \hat{S}^z_{\mathbf{k}\beta} \rangle$

$\langle \hat{c}^\dagger_{\mathbf{0}\alpha\sigma} \hat{c}_{\mathbf{0}\alpha\sigma} \hat{c}^\dagger_{\mathbf{R}\beta\sigma'} \hat{c}_{\mathbf{R}\beta\sigma'} \rangle$ with $\mathbf{R}$ ranging on the all unit cell, (α, β) ranging on the all orbitals in the unit cell, and (σ, σ') ranging from $\uparrow$ to $\downarrow$.

- $\langle \hat{S}^+_{\mathbf{k}\alpha} \hat{S}^-_{\mathbf{k}\beta} \rangle$ and $\langle \hat{\mathbf{S}}_{\mathbf{k}\alpha} \cdot \hat{\mathbf{S}}_{\mathbf{k}\beta} \rangle$

  For HPhi++, $\langle \hat{c}^\dagger_{\mathbf{0}\alpha\sigma} \hat{c}_{\mathbf{0}\alpha-\sigma} \hat{c}^\dagger_{\mathbf{R}\beta-\sigma} \hat{c}_{\mathbf{R}\beta\sigma} \rangle$ with $\mathbf{R}$ ranging on the all unit cell, (α, β) ranging on the all orbitals in the unit cell, and σ ranging from $\uparrow$ to $\downarrow$. For mVMC, $\langle \hat{c}^\dagger_{\mathbf{0}\alpha\sigma} \hat{c}_{\mathbf{R}\beta\sigma} \hat{c}^\dagger_{\mathbf{R}\beta-\sigma} \hat{c}_{\mathbf{0}\alpha-\sigma} \rangle$ with $\mathbf{R}$ ranging on the all unit cell, (α, β) ranging on the all orbitals in the unit cell, and σ ranging from $\uparrow$ to $\downarrow$. In the both cases, please care the order of operators.

In the default settings of Standard mode (`outputmode="corr"`), the above indices are specified automatically. Therefore we do not have to care it.

Results of correlation function in the site representation

The correlation functions having the indices specified in *Specify the index of correlation function to be computed* are computed by mVMC/HPhi++, and written to files. The general description of this file is written in the manuals of mVMC/HPhi++. File names in the *Tutorial* are `output/zvo_cisajs_001.dat` and `output/zvo_cisajscktalt_001.dat` (mVMC), or `output/zvo_cisajs.dat` and `output/zvo_cisajscktalt.dat` (HPhi++).

The utility `fourier` reads these files before the calculation. If some of the correlation functions with indices written in *Specify the index of correlation function to be computed* are lacking (for example, because Standard mode was not used), this utility assume them as 0.

Correlation functions on the *k* path

This file contains the Fourier-transformed correlation function and generated by the utility `fourier`. The file name in the *Tutorial* is `output/zvo_corr.dat`.

```
# k-length[1]
# Orbital   1 to Orbital   1
#  UpUp[   2,    3] (Re. Im.) DownDown[    4,    5]
#  Density[    6,    7] SzSz[    8,    9] S+S-[  10,   11] S.S[  12,   13]
0.00000E+00     0.88211E+00   -0.50000E-09     0.88211E+00     0.40000E-09 ...
0.25000E-01     0.87976E+00   -0.46625E-09     0.87976E+00     0.42882E-09 ...
0.50000E-01     0.87276E+00   -0.42841E-09     0.87276E+00     0.45201E-09 ...
:                                                                      :
```

First, the information of the quantities at each column is written, and then the *k* coordinate along the path and the real- and imaginary- part of the correlation function are written.

gnuplot script

This file is generated by `greenr2k`. This script is used for displaying the *k* labels in gnuplot. The file name is `kpath.gp`.

```
set xtics ('G'     0.00000, 'X'     0.50000, 'M'     1.00000, 'G'    1.70711)
set ylabel 'Correlation function'
set grid xtics lt 1 lc 0
```

FermiSurfer file to display the isosurface of the momentum distribution

This file is generated by `greenr2k`. The file name in the tutorial is `output/zvo_corr_eigen0.dat.frmsf`.

.1.4 Behavior of `greenr2k` utility

This utility is used as follows:

```
$ ${PATH}/greenr2k ${NAMELIST} ${GEOMETRY}
```

where `${PATH}` is the path to the directory where the executable `fourier` exists, ${NAMELIST} is the NameList input-file name of HPhi++/mVMC, and ${GEOMETRY} is the path to the *Geometry* file.

The behavior of this utility is slightly different between the correlation functions from each mode of HPhi++ (Lanczos, TPQ, Full diagonalization, LOBCG) and mVMC. In the following cases, we assume that `CDataFileHead` in the ModPara input file is `"zvo"` (default).

HPhi-Lanczos

In this case, `HPhi` writes correlation functions to the files `zvo_cisajs.dat` (one body) and `zvo_cisajscktalt.dat` (two body) in `output/` directory. `fourier` utility reads this files, performs the Fourier transformation, and generate single file `zvo_corr.dat` in `output/` directory.

HPhi-TPQ

`HPhi` writes correlation functions to files `zvo_cisajs_run*step*.dat` (one body), `zvo_cisajscktalt_run*step*.dat` (two body) at each trial and TPQ step to the `output/` directory. `fourier` utility reads the one- and the two-body correlation function at each trial/TPQ-step, and performs Fourier transformation, and write to a file `zvo_corr_run*step*.dat` in `output/` directory.

HPhi-Full diagonalization and LOBCG

`HPhi` writes correlation functions to files `zvo_cisajs_eigen*.dat` (one body) and `zvo_cisajscktalt_eigen*.dat` (two body) for each wavefunction to the `output/` directory. `fourier` utility reads the one- and the two-body correlation function at each state and performs Fourier transformation, and write to a file `zvo_corr_eigen*.dat` in `output/`.

mVMC

`vmc.out` performs calculations according to the input parameters `NDataIdxStart` and `NDataQtySmp` in `ModPara` file, and it generates `zvo_cisajs_???.dat` (one body) and `zvo_cisajscktalt_???.dat` (two body) in `output/` directory. `fourier` utility reads all of these files, performs Fourier transformation, computes the average

$$\langle A \rangle = \frac{1}{N_{\rm Try}} \sum_{i=1}^{N_{\rm Try}} A_i \tag{9}$$

and the standard error

$$\delta A = \frac{1}{N_{\rm Try} - 1} \sqrt{\frac{1}{N_{\rm Try}} \sum_{i=1}^{N_{\rm Try}} (A_i - \langle A \rangle)^2} \tag{10}$$

of the real- and imaginary-part of each correlation function, and writes them to a file `zvo_corr_eigen*.dat` in `output/` directory.

.1.5 Contact

If you have any comments, questions, bug reports etc. about this utility, please contact to the main developer (Mitsuaki Kawamura) by sending the e-mail (the address is shown below).

```
mkawamura_at_issp.u-tokyo.ac.jp
```

Please change `_at_` into `@`, when you will send the e-mail.

.2 Downfolding with Wannier functions

.2.1 Overview

In this document, we introduce how we compute downfolded models with mVMC or HPhi++ in conjunction to RESPACK.

$$\begin{aligned}
\mathcal{H} = &\sum_{R,R',i,j,\sigma} \left(t_{(R'-R)ij} - t^{\rm DC}_{(R'-R)ij}\right) c^\dagger_{R'j\sigma} c_{Ri\sigma} \\
&+ \sum_{R,i} U_{0ij} n_{Ri\uparrow} n_{Ri\downarrow} + \sum_{(R,i)<(R',j)} U_{(R'-R)ij} n_{Ri} n_{R'j} - \sum_{(R,i)<(R',j)} J_{(R'-R)ij} (n_{Ri\uparrow} n_{R'j\uparrow} + n_{Ri\downarrow} n_{R'j\downarrow}) \\
&+ \sum_{(R,i)<(R',j)} J_{(R'-R)ij} (c^\dagger_{Ri\uparrow} c^\dagger_{R'j\downarrow} c_{Ri\downarrow} c_{R'j\uparrow} + c^\dagger_{R'j\uparrow} c^\dagger_{Ri\downarrow} c_{R'j\downarrow} c_{Ri\uparrow}) \\
&+ \sum_{(R,i)<(R',j)} J_{(R'-R)ij} (c^\dagger_{Ri\uparrow} c^\dagger_{Ri\downarrow} c_{R'j\downarrow} c_{R'j\uparrow} + c^\dagger_{R'j\uparrow} c^\dagger_{R'j\downarrow} c_{Ri\downarrow} c_{Ri\uparrow}), \\
t^{\rm DC}_{0ii} \equiv &\frac{1}{2} U_{0ii} D_{0ii} + \sum_{(R,j)(\neq 0,i)} U_{Rij} D_{0jj} - \frac{1}{2} \sum_{(R,j)(\neq 0,i)} J_{Rij} D_{0jj}, \\
t^{\rm DC}_{Rij} \equiv &\frac{1}{2} J_{Rij} (D_{Rij} + 2{\rm Re}[D_{Rij}]) - \frac{1}{2} U_{Rij} D_{Rij}, \quad (R,j) \neq (0,i), \\
D_{Rij} \equiv &\sum_\sigma \left\langle c^\dagger_{Rj\sigma} c_{0i\sigma} \right\rangle_{\rm KS}.
\end{aligned}$$

Prerequisite

We compute the Kohn-Sham orbitals with QuantumESPRESSO or xTAPP, and obtain the Wannier function, the dielectric function, the effective interaction with RESPACK, and simulate quantum lattice models with mVMC or HPhi++. Therefore, these programs must be available in our machine.

.2.2 Tutorial

In this tutorial, we downfold Sr_2VO_4 into three-orbitals 2D Hubbard model, and simulate that model with HPhi/mVMC. We employ QuantumESPRESSO for the DFT calculation.

SCF calculation of charge density

First, we perform the SCF calculation of the charge density. The input file is as follows:

`scf.in`

```
&CONTROL
 calculation = 'scf'
  pseudo_dir = '../pseudo/'
      prefix = 'sr2vo4'
/
&SYSTEM
       ibrav = 0
         nat = 7
        ntyp = 3
     ecutwfc = 45.000000
     ecutrho = 360.000000
 occupations = 'tetrahedra_opt'
/
&ELECTRONS
mixing_beta = 0.3
/
CELL_PARAMETERS angstrom
 -1.917800 1.917800 6.280100
 1.917800 -1.917800 6.280100
 1.917800 1.917800 -6.280100
ATOMIC_SPECIES
 O 15.999400 O_pbe_v1.2.uspp.F.UPF
 Sr 87.620000 Sr.pbe-spn-rrkjus_psl.1.0.0.UPF
 V 50.941500 V_pbe_v1.uspp.F.UPF
ATOMIC_POSITIONS crystal
 Sr 0.644600 0.644600 1.000000
 Sr 0.355400 0.355400 0.000000
 V 0.000000 0.000000 0.000000
 O 0.500000 0.000000 0.500000
 O 0.000000 0.500000 0.500000
 O 0.842000 0.842000 1.000000
 O 0.158000 0.158000 0.000000
K_POINTS automatic
 4 4 4 0 0 0
```

The pseudopotential (UPF file) are downloaded from The SG15 Optimized Norm-Conserving Vanderbilt (ONCV) pseudopotentials.

http://www.quantum-simulation.org/potentials/sg15_oncv/sg15_oncv_upf_2015-10-07.tar.gz

We use the program `pw.x` in QuantumESPRESSO as follows.

```
$ pw.x -in scf.in
```

(Optional) Band structure

`band.in`

```
&CONTROL
 calculation = 'bands'
  pseudo_dir = '../pseudo/'
```

(continues on next page)

(continued from previous page)

```
      prefix = 'sr2vo4'
/
&SYSTEM
       ibrav = 0
         nat = 7
        ntyp = 3
     ecutwfc = 45.000000
     ecutrho = 360.000000
        nbnd = 57
/
&ELECTRONS
/
CELL_PARAMETERS angstrom
 -1.917800 1.917800 6.280100
 1.917800 -1.917800 6.280100
 1.917800 1.917800 -6.280100
ATOMIC_SPECIES
 O 15.999400 O_pbe_v1.2.uspp.F.UPF
 Sr 87.620000 Sr.pbe-spn-rrkjus_psl.1.0.0.UPF
 V 50.941500 V_pbe_v1.uspp.F.UPF
ATOMIC_POSITIONS crystal
 Sr 0.644600 0.644600 1.000000
 Sr 0.355400 0.355400 0.000000
 V 0.000000 0.000000 0.000000
 O 0.500000 0.000000 0.500000
 O 0.000000 0.500000 0.500000
 O 0.842000 0.842000 1.000000
 O 0.158000 0.158000 0.000000
K_POINTS crystal
          49
        0.5000000000    0.5000000000   -0.5000000000    1.0
        0.4433287500    0.5566712500   -0.5000000000    1.0
        0.3866575000    0.6133425000   -0.5000000000    1.0
        0.3299862500    0.6700137500   -0.5000000000    1.0
        0.2733150000    0.7266850000   -0.5000000000    1.0
        0.2166437500    0.7833562500   -0.5000000000    1.0
        0.1599725000    0.8400275000   -0.5000000000    1.0
        0.1033012500    0.8966987500   -0.5000000000    1.0
        0.0466300000    0.9533700000   -0.5000000000    1.0
        0.0919660000    0.9080340000   -0.4546620000    1.0
        0.1373020000    0.8626980000   -0.4093240000    1.0
        0.1826380000    0.8173620000   -0.3639860000    1.0
        0.2279740000    0.7720260000   -0.3186480000    1.0
        0.2733100000    0.7266900000   -0.2733100000    1.0
        0.3186480000    0.6813520000   -0.3186480000    1.0
        0.3639860000    0.6360140000   -0.3639860000    1.0
        0.4093240000    0.5906760000   -0.4093240000    1.0
        0.4546620000    0.5453380000   -0.4546620000    1.0
        0.5000000000    0.5000000000   -0.5000000000    1.0
        0.3750000000    0.3750000000   -0.3750000000    1.0
        0.2500000000    0.2500000000   -0.2500000000    1.0
        0.1250000000    0.1250000000   -0.1250000000    1.0
        0.0000000000    0.0000000000    0.0000000000    1.0
       -0.0625000000    0.0625000000    0.0000000000    1.0
       -0.1250000000    0.1250000000    0.0000000000    1.0
       -0.1875000000    0.1875000000    0.0000000000    1.0
       -0.2500000000    0.2500000000    0.0000000000    1.0
```

(continues on next page)

(continued from previous page)

```
       -0.3125000000    0.3125000000    0.0000000000    1.0
       -0.3750000000    0.3750000000    0.0000000000    1.0
       -0.4375000000    0.4375000000    0.0000000000    1.0
       -0.5000000000    0.5000000000    0.0000000000    1.0
       -0.5000000000    0.5000000000    0.0466300000    1.0
       -0.4546620000    0.4546620000    0.0919660000    1.0
       -0.4093240000    0.4093240000    0.1373020000    1.0
       -0.3639860000    0.3639860000    0.1826380000    1.0
       -0.3186480000    0.3186480000    0.2279740000    1.0
       -0.2733100000    0.2733100000    0.2733100000    1.0
       -0.2277583333    0.2277583333    0.2277583333    1.0
       -0.1822066667    0.1822066667    0.1822066667    1.0
       -0.1366550000    0.1366550000    0.1366550000    1.0
       -0.0911033333    0.0911033333    0.0911033333    1.0
       -0.0455516667    0.0455516667    0.0455516667    1.0
        0.0000000000    0.0000000000    0.0000000000    1.0
        0.0000000000    0.0833333333    0.0000000000    1.0
        0.0000000000    0.1666666667    0.0000000000    1.0
        0.0000000000    0.2500000000    0.0000000000    1.0
        0.0000000000    0.3333333333    0.0000000000    1.0
        0.0000000000    0.4166666667    0.0000000000    1.0
        0.0000000000    0.5000000000   -0.0000000000    1.0
```

We use `pw.x`.

```
$ pw.x -in band.in
```

`bands.in`

```
&BANDS
      prefix = 'sr2vo4'
       lsym = .false.
/
```

We use `bands.x` QuantumESPRESSO.

```
$ bands.x -in bands.in
```

We can plot the band structure by reading output `bands.out.gnu` from GnuPlot etc.

Kohn-Sham orbitals for Wannier

`nscf.in`

```
&CONTROL
 calculation = 'nscf'
  pseudo_dir = '../pseudo/'
  wf_collect = .true.
      prefix = 'sr2vo4'
/
&SYSTEM
       ibrav = 0
         nat = 7
        ntyp = 3
     ecutwfc = 45.000000
```

(continues on next page)

(continued from previous page)

```
     ecutrho = 360.000000
 occupations = 'tetrahedra_opt'
        nbnd = 57
/
&ELECTRONS
/
CELL_PARAMETERS angstrom
 -1.917800 1.917800 6.280100
 1.917800 -1.917800 6.280100
 1.917800 1.917800 -6.280100
ATOMIC_SPECIES
 O 15.999400 O_pbe_v1.2.uspp.F.UPF
 Sr 87.620000 Sr.pbe-spn-rrkjus_psl.1.0.0.UPF
 V 50.941500 V_pbe_v1.uspp.F.UPF
ATOMIC_POSITIONS crystal
 Sr 0.644600 0.644600 1.000000
 Sr 0.355400 0.355400 0.000000
 V 0.000000 0.000000 0.000000
 O 0.500000 0.000000 0.500000
 O 0.000000 0.500000 0.500000
 O 0.842000 0.842000 1.000000
 O 0.158000 0.158000 0.000000
K_POINTS automatic
 4 4 4 0 0 0
```

We use `pw.x` as

```
$ pw.x -in nscf.in
```

Then, we use the utility `qe2respack.sh` which is included in the RESPACK package. The command-line argument is the name of `[prefix].save` directory.

```
$ qe2respack.sh sr2cuo3.save
```

Wannier function, dielectric function, effective interaction

`respack.in`

```
&PARAM_CHIQW
Num_freq_grid = 1
!Ecut_for_eps =
flg_cRPA = 1
MPI_num_proc_per_qcomm = 1
MPI_num_qcomm = 4
!flg_calc_type = 2
!n_calc_q = 8
/
&PARAM_WANNIER
N_wannier = 3
Lower_energy_window = 9.865
Upper_energy_window = 13.35
N_initial_guess = 3
/
dxy 0.2 0.0 0.0 0.0
dyz 0.2 0.0 0.0 0.0
```

(continues on next page)

(continued from previous page)

```
dzx 0.2 0.0 0.0 0.0
&PARAM_INTERPOLATION
N_sym_points = 10
dense = 12, 12, 12
/
0.50000 0.50000 -0.50000
0.04663 0.95337 -0.50000
0.27331 0.72669 -0.27331
0.50000 0.50000 -0.50000
0.00000 0.00000 0.00000
-0.50000 0.50000 0.00000
-0.50000 0.50000 0.04663
-0.27331 0.27331 0.27331
0.00000 0.00000 0.00000
0.00000 0.50000 0.00000
&PARAM_VISUALIZATION
flg_vis_wannier = 1,
ix_vis_min = -1,
ix_vis_max = 2,
iy_vis_min = -1,
iy_vis_max = 2,
iz_vis_min = -1,
iz_vis_max = 2
/
&PARAM_CALC_INT
calc_ifreq = 1
ix_intJ_min = 0
ix_intJ_max = 0
iy_intJ_min = 0
iy_intJ_max = 0
iz_intJ_min = 0
iz_intJ_max = 0
/
```

We use `calc_wannier`, `calc_chiqw`, `calc_j3d`, `calc_w3d` in RESPACK.

```
$ calc_wannier < respack.in
$ calc_chiqw < respack.in
$ calc_w3d < respack.in
$ calc_j3d < respack.in
```

Quantum lattice mode for HPhi/mVMC

First, we translate the hopping file etc. created by RESPACK into the Wannier90 format. For this purpose, we use the utility `respack2wan90.py` included in HPhi/mVMC. The command-line argument should be the same as the input parameter `CDataFileHead` for Standard mode of HPhi/mVMC. If we do not specify that argument, the default value `zvo` is used.

```
$ respack2wan90.py zvo
```

Then we can run HPhi/mVMC with the standard mode.

`respack.in`

```
model = "Hubbard"
lattice = "wannier90"
a0w = 2
a0l = 0
a0h = 2
a1w = 0
a1l = 2
a1h = 2
a2w = 1
a2l = 0
a2h = 0
method = "CG"
2Sz = 0
nelec = 4
exct = 1
cutoff_t = 0.2
cutoff_u = 0.4
cutoff_j = 0.1
```

```
$ vmc.out -s stan.in
```

.2.3 Input parameter for Standard mode

We show the following example of the input file.

```
model = "Hubbard"
lattice = "wannier90"
a0w = 2
a0l = 0
a0h = 2
a1w = 0
a1l = 2
a1h = 2
a2w = 1
a2l = 0
a2h = 0
method = "CG"
2Sz = 0
nelec = 4
exct = 1
cutoff_t = 0.2
cutoff_u = 0.4
cutoff_j = 0.1
```

The input parameters for the Standard mode to perform calculation of the downfolded model are as follows:

- `lattice = "wannier90"`
- `cutoff_t`, `cutoff_u`, `cutoff_j`

  Type : float

  Default : `1.0e-8`

  The cutoff for the hopping, Coulomb, exchange integrals. We ignore these integrals smaller than cutoffs.

- `cutoff_tW`, `cutoff_tL`, `cutoff_tH`

- `cutoff_UW`, `cutoff_UL`, `cutoff_UH`
- `cutoff_JW`, `cutoff_JL`, `cutoff_JH`

  Type : Integer

  Default : Including all range

  The cutoff for the hopping, Coulomb, exchange integrals. We ignore these integrals that have lattice vector $\mathbf{R}$ larger than these values.

- `cutoff_length_t`, `cutoff_length_U`, `cutoff_length_J`

  Type : float

  Default : -1.0 (Including all range)

  The cutoff for the hopping, Coulomb, exchange integrals. We ignore these integrals whose distance is longer than this value. Its distance id computed with the position of the Wannier center and unit lattice vectors.

- `W`, `L`, `Height`
- `a0W`, `a0L`, `a0H`, `a1W`, `a1L`, `a1H`, `a2W`, `a2L`, `a2H`
- `Wsub`, `Lsub`, `Hsub`
- `a0Wsub`, `a0Lsub`, `a0Hsub`, `a1Wsub`, `a1Lsub`, `a1Hsub`, `a2Wsub`, `a2Lsub`, `a2Hsub`

  The third dimension appears.

.2.4 File format

Standard mode of HPhi/mVMC reads the following files. We can obtain these files by executing the utility `respack2wan90.py` in the directory where the programs of RESPACK are executed. `respack2wan90.py` is included in HPhi/mVMC package.

Geometry

The file name is `[CDataFileHeat]_geom.dat`. By editing this file, we can modify the number of orbitals treated in HPhi/mVMC.

```
-1.917800 1.917800 6.280100
1.917800 -1.917800 6.280100
1.917800 1.917800 -6.280100
3
0.000000 -0.000000 -0.000000
-0.000000 -0.000000 -0.000000
0.000000 0.000000 0.000000
```

- Lines 1 - 3

  Unit lattice vectors in the Cartesian coordinate (arbitrary unit).

- Line 4

  The number of orbitals par unit cell treated by mVMC/HPhi. When this file is generated by `respack2wan90.py`, this number is the same as the number of Wannier functions in RESPACK. When we reduce the number by editing this file, the model including the same number of orbitals from the top.

- Line 5 - end

  Wannier centers in the fractional coordinate. They are used by the Fourier utility.

Hopping, Coulomb, exchange integrals

The file name is `[CDataFileHeat]_hr.dat`, `[CDataFileHeat]_ur.dat`, and `[CDataFileHeat]_jr.dat`, respectively. They are formatted as the hopping-integral file of Wannier90 is used.

.2.5 Contact

If you have any comments, questions, bug reports etc. about this utility, please contact to the main developer (Mitsuaki Kawamura) by sending the e-mail (the address is shown below).

```
mkawamura_at_issp.u-tokyo.ac.jp
```

Please change `_at_` into `@`, when you will send the e-mail.

.3 Acknowledgement

We would like to express our sincere gratitude to Prof. Hidetoshi Nishimori and Mr. Daisuke Tahara. Implementation of the Lanczos algorithm in HPhi++ written in C is based on the pioneering diagonalization package TITPACK written in Fortran by Prof. Nishimori. For developing the user interface of HPhi++, we followed 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 HPhi++ is based on his original codes. We would also like to express our thanks for the support of the "*Project for advancement of software usability in materials science*" of The Institute for Solid State Physics, The University of Tokyo, for the development of HPhi++ ver. 0.1 - ver. 2.0.4

Dr. Yuichi Motoyama and Dr. Yasuyuki Kato found and reported bugs. We acknowledge their kindness and carefulness.

We thank Prof. Takeo Hoshi and Prof. Tomohiro Sogabe for the fruitful discussions on the shifted Krylov method and the development of the $K\omega$ library.