Manual for mVMC
ver. 0.1

October 26, 2016

Contents

1 What is mVMC?

High-accuracy analyses of theoretical models for quantum many-body systems are expected to play important role for clarifying the nature of novel quantum phases such high-temperature superconductivities and quantum spin liquids. Furthermore, recent theoretical progress enables us to obtain low-energy effective models for real materials with non-empirical way [1]. To clarify the electronic structures of real materials and control the physical properties, it is important perform the high-accuracy analyses the low-energy effective models. One of the most reliable theoretical tools for treating the strongly correlated electron systems is the exact diagonalization method. However, applicable range of system size is strongly limited in the exact diagonalization method. Variational Monte Carlo method [2] is one of promising way to perform the high-accuracy calculations for the larger system sizes beyond exact diagonalization method. Although the strong limitation of the variational wave function is the origin of the poor accuracy of the variational Monte Carlo method, recent development of the theoretical mehtod and computers relaxes the limitation of the variational wave functions and enable us to perform the high-accuracy calculations [3–5].

mVMC(many-variable variational Monte Carlo method) is a software for performing the high-accurate variational Monte Carlo calculations with the simple and flexible user interface. mVMC also supports the massively parallel calculations. For the conventional model in strongly correlated electron systems such as the Hubbard model, the Heisenberg model, and the Kondo-lattice model, users can perform the calculation by preparing the one input files whose length is shorter the ten lines. By using the same input file, user can perform the exact diagonalization through $\mathcal{H}\Phi$ [6]. Thus, it is easy to perform the calculations for large system sizes that can not be treated by the exact diagonalization after examining the accuracy of the variational calculation for small system sizes. A broad spectrum of users including experimental scientists and scientists of quantum chemistry is cordially welcome.

1.1 Overview of mVMC

By using mVMC, the following calculation can be done:

- The variational wave function which gives the minimum value of the expected value of energy in the range of the degree of freedoms of variational parameters is numerically generated. The calculation limited to the partial space divided by quantum numbers is also possible.

- The expected values of the physical quantities such as correlation functions can be calculated by using the generated variational wave functions.

The calculation flow in mVMC is shown as follows:

1. Read input files (*.def)
2. Optimize variational parameters $\boldsymbol{\alpha}$ to minimize $\langle \mathcal{H} \rangle$
3. Calculate one body and two body Green functions
4. Output variational parameters and expected values

In calculation, the simple parallelization for the generation of the real space arrangement $|x\rangle$ and the collecting samples and the calculation result of expected energies is implemented. Following the procedure for each cluster computers, the parallelized calculation using MPI is automatically done by indicating the parallel number. However, mVMC cannot execute under the environment where the MPI job is forbidden such as the front-end of system B at ISSP. In mVMC, we use PFAPACK [7] to compute the Pfaffian matrix.

1.2 License

The distribution of the program package and the source codes for mVMC follows GNU General Public License version 3 (GPL v3).

1.3 Copyright

©2016 Takahiro Misawa, Satoshi Morita, Takahiro Ohgoe, Kota Ido, Mitsuaki Kawamura, Takeo Kato, Masatoshi Imada. All rights reserved.

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

1.4 Contributors

This software is developed by following contributors.

- ver.0.1 (released at 2016/10/26)
  - Developers
    - Takahiro Misawa
      (The Institute for Solid State Physics, The University of Tokyo)
    - Satoshi Morita
      (The Institute for Solid State Physics, The University of Tokyo)

    - Takahiro Ogoe
      (Department of Applied Physics, The University of Tokyo)
    - Kota Ido
      (Department of Applied Physics, The University of Tokyo)
    - Masatoshi Imada
      (Department of Applied Physics, The University of Tokyo)
    - Mitsuaki Kawamura
      (The Institute for Solid State Physics, The University of Tokyo)
    - Kazusyohi Yoshimi
      (The Institute for Solid State Physics, The University of Tokyo)
  - Project coordinator
    - Takeo Kato
      (The Institute for Solid State Physics, The University of Tokyo)

1.5 Operating environment

mVMC is tested in the following platform:

- The supercomputer system-B “sekirei” and system-C “maki” in ISSP
- K computer
- OpenMPI + Intel Compiler + MKL
- MPICH + Intel Compiler + MKL
- MPICH + GNU Compiler + MKL

2
How to use mVMC?

2.1 Prerequisite

mVMC requires the following packages:

- C compiler (intel, Fujitsu, GNU, etc.)
- ScaLAPACK library (intel MKL, Fujitsu, ATLAS, etc.)
- MPI library

Tips

E. g. / Settings of intel compiler

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

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

or

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

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

2.2 Installation

You can download mVMC in the following place.

You can obtain the mVMC directory by typing

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

There are two kind of procedures to install mVMC.

2.2.1 Using `config.sh`

Please run `config.sh` script in the mVMC directory as follow (for ISSP system-B "sekirei"):

```
$ bash config.sh sekirei
```

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

- `sekirei` : ISSP system-B "sekirei"
- `kei` : K computer and ISSP system-C "maki"
- `openmpi-intel` : OpenMPI + intel compiler
- `mpich-intel` : MPICH + intel compiler
- `mpich-gnu-mkl` : MPICH + GNU Compiler + MKL
- `gnu` : OpenMPI + GCC

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

```
CC = mpicc
LIB = -L$(MKLROOT)/lib/intel64 -lmkl_scalapack_lp64 -lmkl_intel_lp64 \
      -lmkl_intel_thread -lmkl_core -lmkl_blacs_sgimpt_lp64 -lpthread -lm
CFLAGS = -O3 -no-prec-div -xHost -qopenmp -Wno-unknown-pragmas
REPORT = -qopt-report-phase=openmp -qopt-report-phase=par
OPTION = -D_mpi_use
CP = cp -f -v
AR = ar rv
FORT = ifort
FFLAGS = -O3 -implicitnone -xHost
SMFTFLAGS = -O3 -no-ansi-alias -xHost -DMEXP=19937 -DHAVE_SSE2
```

We explain macros of this file as:

- `CC` : The C Compilation command (`mpicc`, `mpifccpx`)
- `Lib` : Compilation options for ScaLAPACK.
- `CFLAGS` : Other compilation options.
- `FORT` : Fortran compilation command (`ifort`, `frtpx`)

Then you are ready to compile mVMC. Please type

```
$ make mvmc
```

and obtain `vmc.out` and `vmcdry.out` in `src/` directory; you should add this directory to the `$PATH`.

Tips

You can make a PATH to mVMC as follows:
`$ export PATH=${PATH}:`*mVMC_top_directory*`/src/`
If you keep this PATH, you should write above in `~/.bashrc` (for `bash` as a login shell)

2.2.2 Using `cmake`

Tips

Before using cmake for sekirei, you must type

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

while for maki, you must type

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

We can compile mVMC as

```
cd $HOME/build/mvmc
cmake -DCONFIG=gcc $PathTomVMC
make
```

Here, we set a path to mVMC as `$PathTomVMC` and to a build directory as `$HOME/build/mvmc`. After compiling, a src folder is constructed below a `$HOME/build/mvmc` folder and obtain executables `vmc.out` and `vmcdry.out` in `src/` directory.

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

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

An example for compiling mVMC by an intel compiler is shown as follows,

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

After compiling, a `src` folder is made below the `build` folder and executes `vmc.out` and `vmcdry.out` are made in the `src` folder. It is noted that we must delete the `build` folder and do the above works again when we change the compilers.

2.3 Directory structure

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

```
├──── COPYING
├──── config.sh
├──── doc/
│      ├──── bib/
│      │      ├──── elsart-num_mod.bst
│      │      └──── userguide.bib
│      ├──── figs/
│      │      ├────*.pdf
│      │      └────*.xbb
│      ├──── jp/
│      │      └────*.tex
│      └──── en/
│            └────*.tex
├──── sample/
│      └──── Standard/
│                ├──── Hubbard/
│                │      ├── square/
│                │      │      ├──── StdFace.def
│                │      │      └──── reference/
│                │      │              └────**.dat
│                │      └── triangular/
│                │           └──── ...
│                ├──── Kondo/
│                │      └── chain/
│                │           └──── ...
│                └──── Spin/
│                         ├── HeisenbergChain/
│                         │      └──── ...
│                         ├── HeisenbergSquare/
│                         │      └──── ...
│                         └── Kagome/
│                               └──── ...
│
└──── src/
        ├────**.c
        ├────**.h
        ├──── ComplexUHF/
        │      └──── src/
        │              ├── **.c
        │              └──── include/
        │                      └────**.h
        │
```

```
├──── makefile_src
├──── include/
│       └────**.h
├──── StdFace/
│       ├────**.c
│       └────**.h
├──── pfapack/
│       ├──── makefile_pfapack
│       └────**.f
└──── sfmt/
        ├──── makefkie_sfmt
        ├────**.c
        └────**.h
```

2.4 Basic usage

mVMC has two executables; one generates detailed input files and the other perform the main calculation with these files. Here, the basic flows of calculations by using these programs.

1. Prepare a minimal input file

   You can choose a model (the Heisenberg model, the Hubbard model, etc.) and a lattice (the square lattice, the triangular lattice, etc.) from ones provided; you can specify some parameters (such as the first/second nearest neighbor hopping integrals, the on-site Coulomb integral, etc.) for them. The input file format is described in the Sec. 4.1.

2. Run

   Run a executable `vmcdry.out` in terminal by specifing the name of input file written in previous step. MPI is not used in this step.

   $ *Path*/`vmcdry.out` *Input_file_name*

   Then, run a executable `vmc.out` with the *namelist* file generated in the previous step.

   `$ mpiexec -np` *number_of_processes* *Path*/`vmcdry.out namelist.def`

   When you use a queuing system in workstations or super computers, sometimes the number of processes is specified as an argument for the job-submitting command. If you need more information, please refer manuals for your system.

3. Watch calculation logs

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

4. Results

   If the calculation is finished normally, the result files are outputted in the current directry. The details of output files are shown in Sec. 4.3.

Tips

The number of threads for OpenMP

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

```
export OMP_NUM_THREADS=16
```

2.5 Printing version ID

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

```
$ PATH/vmcdry.out -v
$ PATH/vmc.out -v
```

3
Tutorial

3.1 List of sample files

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

- The Hubbard model on the two dimensional square lattice
  (`samples/Standard/Hubbard/square/`)
- The Hubbard model on the two dimensional triangular lattice
  (`samples/Standard/Hubbard/triangular/`)
- The one dimensional Kondo chain
  (`samples/Standard/Kondo/chain/`)
- The one dimensional antiferromagnetic Heisenberg chain
  (`samples/Standard/Spin/HeisenbergChain/HeisenbergChain/`)
- The antiferromagnetic Heisenberg model on the two dimensional square lattice
  (`samples/Standard/Spin/HeisenbergSquare/`)
- The antiferromagnetic Heisenberg model on the two dimensional Kagome lattice
  (`samples/Standard/Spin/Kagome/`)

We can perform these tutorials in the same way. In the following, the tutorial of the one dimensional antiferromagnetic Heisenberg chain is shown.

3.2 Heisenberg model

This tutorial should be performed in

`sample/Standard/Spin/HeisenbergChain/`.

This directory contains the following things:
The input file: StdFace.def
reference outputs: reference/

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

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

The input file is as follows:

```
L = 16
Lsub=4
model = "Spin"
lattice = "chain lattice"
J = 1.0
2Sz = 0
NMPtrans=1
```

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

Making detailed input files

In mVMC, the detailed input files should be made as a first step. We first execute the following command.

```
$ Path/vmcdry.out StdFace.def
```

Then, we can see the following standard outputs in the terminal.

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

  Open Standard-Mode Inputfile StdFace.def

  KEYWORD : l                    | VALUE : 16
  KEYWORD : lsub                 | VALUE : 4
  KEYWORD : model                | VALUE : spin
  KEYWORD : lattice              | VALUE : chain
  KEYWORD : j                    | VALUE : 1.0
  KEYWORD : nmptrans             | VALUE : 1

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

  @ Lattice Size & Shape

                L = 16
             Lsub = 4
                L = 16
                W = 1
```

```
          phase0 = 0.00000     ######  DEFAULT VALUE IS USED  ######

  @ Hamiltonian

              2S = 1           ######  DEFAULT VALUE IS USED  ######
               h = 0.00000     ######  DEFAULT VALUE IS USED  ######
           Gamma = 0.00000     ######  DEFAULT VALUE IS USED  ######
               D = 0.00000     ######  DEFAULT VALUE IS USED  ######
             J0x = 1.00000
             J0y = 1.00000
             J0z = 1.00000

  @ Numerical conditions

            Lsub = 4
            Wsub = 1
     ioutputmode = 1           ######  DEFAULT VALUE IS USED  ######

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

    qptransidx.def is written.
         filehead = zvo         ######  DEFAULT VALUE IS USED  ######
         filehead = zqp         ######  DEFAULT VALUE IS USED  ######
      NVMCCalMode = 0           ######  DEFAULT VALUE IS USED  ######
     NLanczosMode = 0           ######  DEFAULT VALUE IS USED  ######
    NDataIdxStart = 1           ######  DEFAULT VALUE IS USED  ######
      NDataQtySmp = 1           ######  DEFAULT VALUE IS USED  ######
      NSPGaussLeg = 8           ######  DEFAULT VALUE IS USED  ######
         NMPTrans = 1
    NSROptItrStep = 1000        ######  DEFAULT VALUE IS USED  ######
     NSROptItrSmp = 100         ######  DEFAULT VALUE IS USED  ######
       NVMCWarmUp = 10          ######  DEFAULT VALUE IS USED  ######
     NVMCInterval = 1           ######  DEFAULT VALUE IS USED  ######
       NVMCSample = 1000        ######  DEFAULT VALUE IS USED  ######
    NExUpdatePath = 2
          RndSeed = 123456789   ######  DEFAULT VALUE IS USED  ######
       NSplitSize = 1           ######  DEFAULT VALUE IS USED  ######
           NStore = 0           ######  DEFAULT VALUE IS USED  ######
     DSROptRedCut = 0.00100     ######  DEFAULT VALUE IS USED  ######
     DSROptStaDel = 0.02000     ######  DEFAULT VALUE IS USED  ######
     DSROptStepDt = 0.02000     ######  DEFAULT VALUE IS USED  ######
          NSPStot = 0           ######  DEFAULT VALUE IS USED  ######
      ComplexType = 0           ######  DEFAULT VALUE IS USED  ######
    locspn.def is written.
    trans.def is written.
    interall.def is written.
    jastrowidx.def is written.
    coulombintra.def is written.
    coulombinter.def is written.
    hund.def is written.
    exchange.def is written.
    orbitalidx.def is written.
    gutzwilleridx.def is written.
    namelist.def is written.
    modpara.def is written.
    greenone.def is written.
    greentwo.def is written.

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

In the beginning of this run, files describing the detail of considered Hamiltonian

- `locspin.def`
- `trans.def`
- `coulombinter.def`
- `coulombintra.def`
- `exchange.def`
- `hund.def`
- `namelist.def`
- `modpara.def`

and files for setting variational parameters

- `gutzwilleridx.def`
- `jastrowidx.def`
- `orbitalidx.def`
- `qptransidx.def`

and files specifying elements of correlation functions that will be calculated

- `greenone.def`
- `greentwo.def`

are generated. The details of these files are shown in Sec. 4.2.

Running

By reading the detailed input files, mVMC executes the calculation of optimizing the variational parameters.

We execute the following command.

`$ mpiexec -np` *number_of_processes* *Path*`/vmcdry.out namelist.def`

The MPI command depends on your system (such as `mpiexec`, `mpirun`, `mpijob`, `poe`, etc.). Then, the calculation starts and the following standard message is outputted in the terminal.

```
-----------
Start: Read *def files.
  Read File namelist.def .
  Read File 'modpara.def' for ModPara.
  Read File 'locspn.def' for LocSpin.
  Read File 'trans.def' for Trans.
  Read File 'coulombintra.def' for CoulombIntra.
  Read File 'coulombinter.def' for CoulombInter.
  Read File 'hund.def' for Hund.
  Read File 'exchange.def' for Exchange.
  Read File 'gutzwilleridx.def' for Gutzwiller.
  Read File 'jastrowidx.def' for Jastrow.
  Read File 'orbitalidx.def' for Orbital.
  Read File 'qptransidx.def' for TransSym.
  Read File 'greenone.def' for OneBodyG.
  Read File 'greentwo.def' for TwoBodyG.
End  : Read *def files.
Start: Read parameters from *def files.
End  : Read parameters from *def files.
Start: Set memories.
End  : Set memories.
Start: Initialize parameters.
End  : Initialize parameters.
Start: Initialize variables for quantum projection.
End  : Initialize variables for quantum projection.
Start: Optimize VMC parameters.
End  : Optimize VMC parameters.
-----------
```

Under the calculation, the following file is outputted:

```
zvo_SRinfo.dat
zvo_out_001.dat
zvo_time_001.dat
zvo_var_001.dat
zvo_CalcTimer.dat
```

In `zvo_out_001.dat`, the following quantities are outputted at each bins

$$\langle H\rangle, \langle H^2\rangle, \frac{\langle H^2\rangle - \langle H\rangle^2}{\langle H\rangle^2}.$$

By seeing these informations, the conversion of the calculation can be judged. By using gnuplot, we can check the evolution of $\langle H\rangle$ as follows:

```
plot "zvo_out_001.dat" u 1
```

The details of these outputted files are shown in Sec. 4.3.

Output results

After finishing calculation normally, the files for the energy, the deviation, the optimized variational parameters and the time of execution for each calculation steps are outputted. In the following, the outputted files are shown

```
gutzwiller_opt.dat
jastrow_opt.dat
orbital_opt.dat
zqp_opt.dat
ClacTimer.dat
```

The details of these outputted files are shown in Sec. 4.3.

Calculation of Green functions

After changing the value of `NVMCCalMode` from 0 to 1 in `modpara.def` file, we execute the following command. When we add `"zqp_opt.dat"` after `"namelist.dat"` as a commandline argument as follows, the calculation of Green functions is done by using the optimized variational parameters.

```
$ Path/vmc.out namelist.def zqp_opt.dat
```

After finishing the calculation, the following files are outputted.

```
zvo_cisajs_001.dat
zvo_cisajscktalt_001.dat
```

The details of these outputted files are shown in Sec. 4.3.

In mVMC, the calculation is done by reading input files categorized by the following six parts.

(1) List: Specify the kinds and names of input files.

(2) Basic parameters: Specify the basic parameters.

(3) Set Hamiltonian: Specify the Hamiltonian.

(4) Set condition of variational parameters : Specify the variational parameters to be optimized.

(5) Initial variational parameters: Specify the initial values of the variational parameters.

(6) Output: Specify the components of one-body and two-body Green's functions to be outputted.

The calculation for complex models can be done by directly making above input files. The details for each files are shown in Sec. 4.2.

4
File specification

4.1 Input files for `vmcdry.out`

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

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

 lattice = "triangular lattice"
//mu = 1.0
// t = -1.0
// t' = -0.5
// U = 8.0
//V = 4.0
//V'=2.0
J = -1.0
J'=-0.5
// nelec = 8
```

Basic rules for input files

- In each line, there is a set of a keyword (before an "=") and a parameter(after an "="); they are separated by "=".
- You can describe keywords in a random order.
- Empty lines and lines beginning in a "`//`"(comment outs) are skipped.
- Upper- and lowercase are not distinguished. Double quotes and blanks are ignored.
- There are three kinds of parameters.
  1. Parameters that must be specified (if not, `vmcdry.out` will stop with error messages),
  2. Parameters that is not necessary be specified (if not, default values are used),
  3. Parameters that must not be specified (if specified, `vmcdry.out` will stop with error messages).
  An example of 3 is transfer t for the Heisenberg spin system. If you choose "model=spin", you should not specify "t".

We explain each keywords as follows:

4.1.1 Parameters about the kind of a calculation

- `model`

  Type : String (Choose from `"Fermion Hubbard"`, `"Spin"`, `"Kondo Lattice"`)

  Description : The target model is specified with this parameter; `"Fermion Hubbard"` denotes the canonical ensemble of the Fermion in the Hubbard model

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

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

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

  `"Kondo Lattice"` denotes canonical ensemble in the Kondo lattice model

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

- `lattice`

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

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

4.1.2 Parameters for the lattice

Chain [Fig. 4.1(a)]

- `L`

  Type : Integer

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

Ladder (Fig. 4.4)

- `L`

  Type : Integer

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

- `W`

  Type : Integer

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

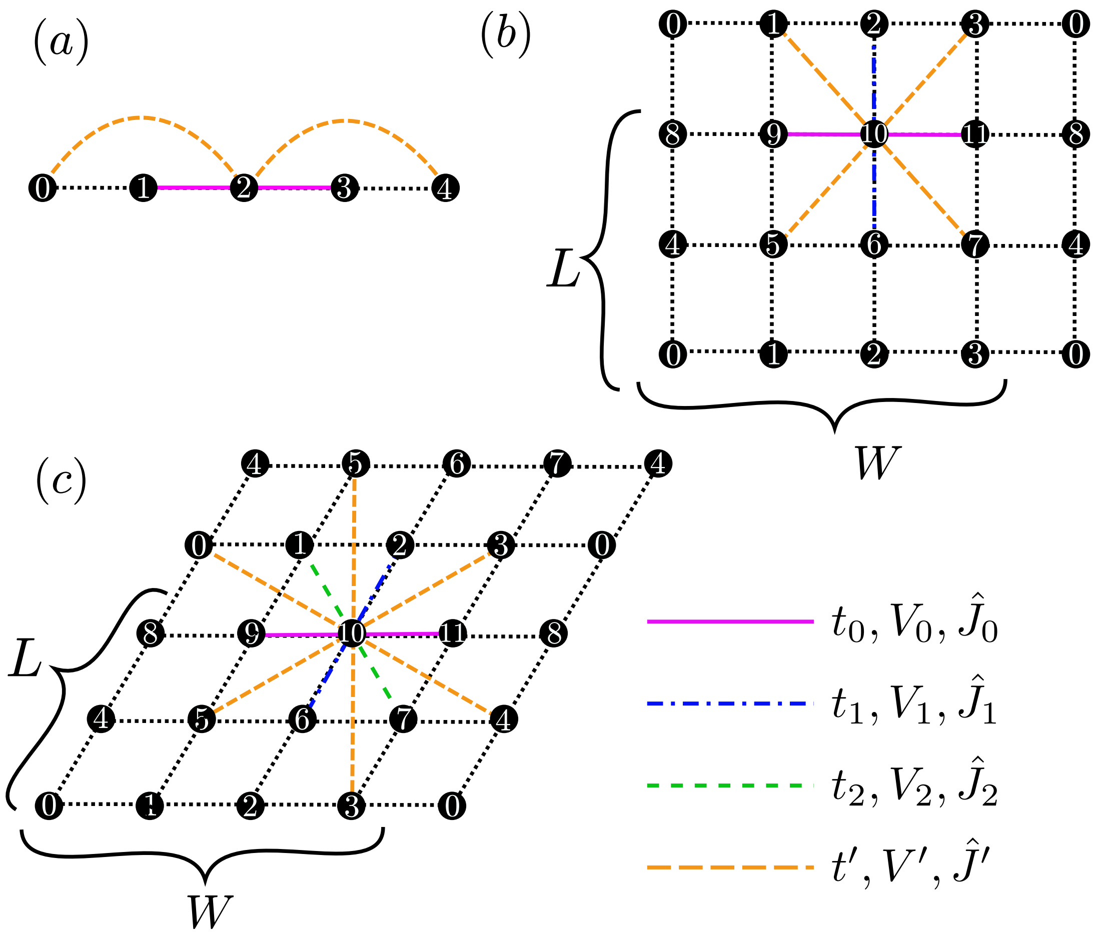

Figure 4.1: Schematic illustration of (a) one dimensional chain lattice, (b) two dimensional square lattice, and (c) two dimensional triangular lattice. They have t, V, and J as a nearest neighbor hopping, an offsite Coulomb integral, and a spin-coupling constant, respectively (magenta solid lines);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

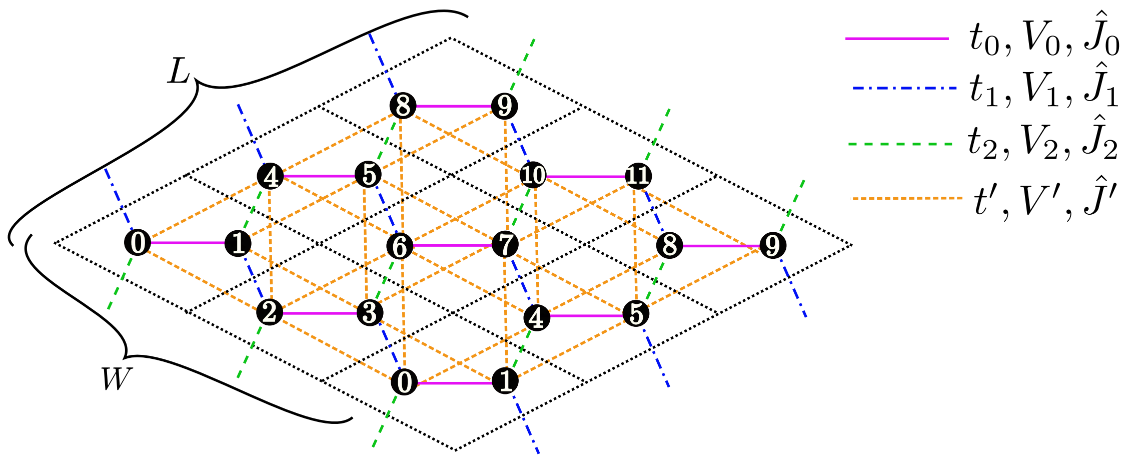

Figure 4.2: Schematic illustration of the anisotropic honeycomb lattice. The nearest neighbor hopping integral, spin coupling, offsite Coulomb integral depend on the bond direction. Those between second nearest neighbor sites are not supported.

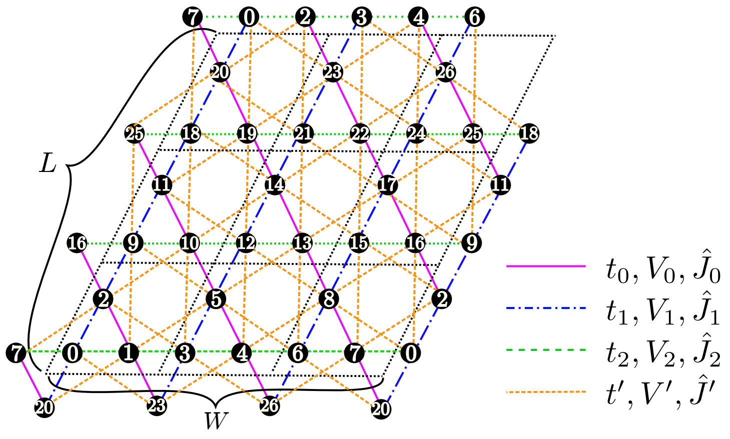

Figure 4.3: Schematic illustration of the Kagome lattice.

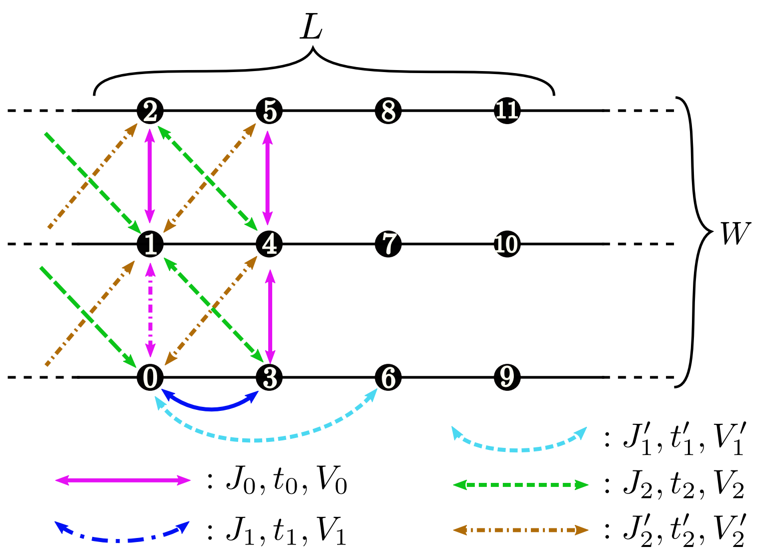

Figure 4.4: Schematic illustration of the ladder lattice.

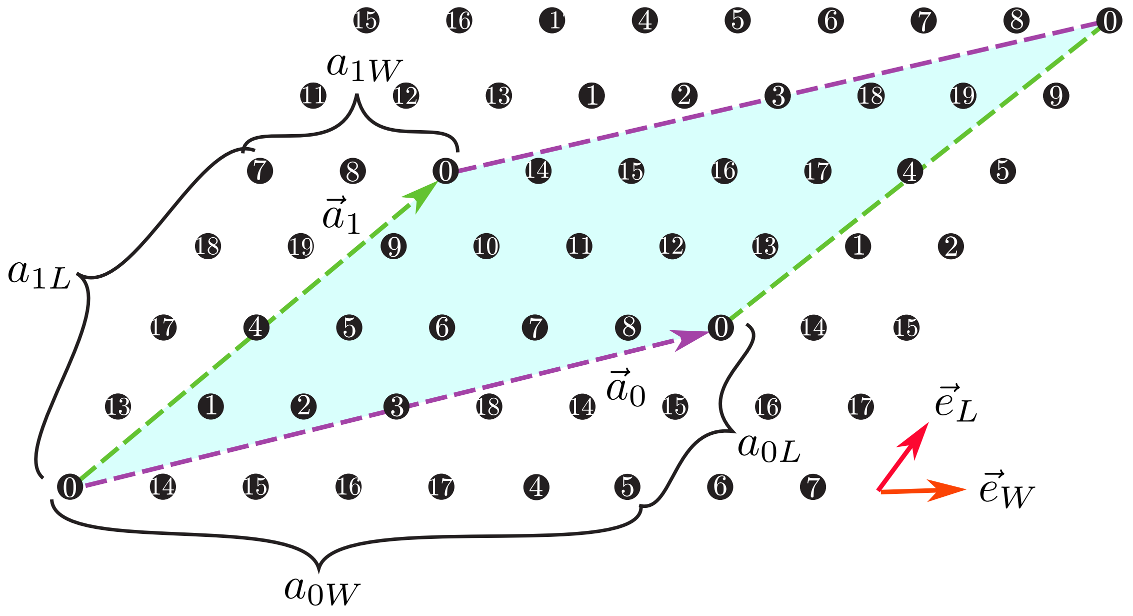

Figure 4.5: The shape of the numerical cell when $\boldsymbol{a}_0 = (6, 2), \boldsymbol{a}_1 = (2, 4)$ in the triangular lattice. The region surrounded by $\boldsymbol{a}_0$(Magenta dashed arrow) and $\boldsymbol{a}_1$(Green dashed arrow) becomes the cell to be calculated (20 sites).

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

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

- `W`, `L`

  Type : Integer

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

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

  Type : Integer

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

If we use both of these method, `vmcdry.out` stops.

We can check the shape of the numerical cell by using a file `lattice.gp`(only for square, trianguler, honeycomb, and kagome lattice) which is written in the Standard mode. This file can be read by `gnuplot` as follows:

```
$ gnuplot lattice.gp
```

4.1.3 Sublattice

By using the following parameters, we can force the pair-orbital symmetrical to the translation of the sublattice.

- `a0Wsub`, `a0Lsub`, `a1Wsub`, `a1Lsub`, `Wsub`, `Lsub`

  Type : Positive integer. In the default setting, `a0Wsub=a0W`, `a0Lsub=a0L`, `a1Wsub=a1W`, `a1Lsub=a1L`, `Wsub=W`, and `Lsub=L`. Namely, there is no sublattice.

  Description : We can specify these parameter as we specify `a0W`, `a0L`, `a1W`, `a1L`, `W`, `L`. If the sublattice is incommensurate with the original lattice, `vmcdry.out` stops.

4.1.4 Parameters for the Hamiltonian

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

Local terms

- `mu`

  Type : Real

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

- `U`

  Type : Real

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

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

  Type : Real

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

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

  Type : Real

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

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

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

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

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

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

  Type : Complex

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

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

  Type : Real

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

- `J0x`, `J0y`, `J0z`, `J0xy`, `J0yx`, `J0xz`, `J0zx`, `J0yz`, `J0zy`
- `J1x`, `J1y`, `J1z`, `J1xy`, `J1yx`, `J1xz`, `J1zx`, `J1yz`, `J1zy`
- `J1'x`, `J1'y`, `J1'z`, `J1'xy`, `J1'yx`, `J1'xz`, `J1'zx`, `J1'yz`, `J1'zy`
- `J2x`, `J2y`, `J2z`, `J2xy`, `J2yx`, `J2xz`, `J2zx`, `J2yz`, `J2zy`
- `J2'x`, `J2'y`, `J2'z`, `J2'xy`, `J2'yx`, `J2'xz`, `J2'zx`, `J2'yz`, `J2'zy`

  Type : Real

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

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

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

  Type : Complex

  Description : (Hubbard and Kondo lattice model) Nearest neighbor hoppings for each direction (See Figs. 4.1-4.3. These bonds are depicted with different line styles.) are specified with these parameter. If there is no bond dependence of the nearest-neighbor 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, `vmcdry.out` will stop.

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

  Type : Real

  Description : (Hubbard and Kondo lattice model) Nearest-neighbor offsite Coulomb integrals V for each direction (See Figs. 4.1-4.3. These bonds are depicted with different line styles.) are specified with these parameters. If there is no bond dependence of the nearest-neighbor 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, `vmcdry.out` will stop.

- `J0x`, `J0y`, `J0z`, `J0xy`, `J0yx`, `J0xz`, `J0zx`, `J0yz`, `J0zy`
- `J1x`, `J1y`, `J1z`, `J1xy`, `J1yx`, `J1xz`, `J1zx`, `J1yz`, `J1zy`
- `J2x`, `J2y`, `J2z`, `J2xy`, `J2yx`, `J2xz`, `J2zx`, `J2yz`, `J2zy`

  Type : Real

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

If there is no bond dependence of the nearest-neighbor 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 counter parts (`J0x`∼`J2zy`), `vmcdry.out` will stop. Below, examples of parameter sets for nearest-neighbor exchange couplings are shown.

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

• `t'`

Type : Complex

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

• `V'`

Type : Real

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

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

Type : Real

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

- `phase0`, `phase1`

  Type : Double complex (`1.0` as defaults)

  Description : We can specify the phase for the prefactor for the hopping through the cell boundary with these parameter. These fuctor 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 $i+1$-th site through the cell boundary with the positive direction becomes as

$$\exp(i \times \text{phase0}) \times t\hat{c}^{\dagger}_{i+1\sigma}\hat{c}_{i\sigma} + \exp(-i \times \text{phase0}) \times t^{*}\hat{c}^{\dagger}_{i\sigma}\hat{c}_{i+1\sigma} \tag{4.4}$$

4.1.5 Parameters for the numerical condition

- `nelec`

  Type : int-type (must be specified)

  Description : The number of itenerant electrons. It is the sum of the $\uparrow$ and $\downarrow$ electrons.

- `NVMCCalMode`

  Type : int-type (default value: 0)

  Description : [0] Optimization of variational parameters, [1] Calculation of one body and two body Green's functions.

- `NDataIdxStart`

  Type : int-type (default value: 1)

  Description : An integer for numbering of output files. For `NVMCCalMode`$=0$, `NDataIdxStart` is added at the end of the output files. For `NVMCCalMode` $=1$, the files are outputted with the number from `NDataIdxStart` to `NDataIdxStart`+`NDataQtySmp`-1.

- `NDataQtySmp`

  Type : int-type (default value: 1)

  Description : The set number for outputted files (only used for `NVMCCalMode` $=$ 1).

- `NSPGaussLeg`

  Type : int-type (Positive integer, default value: 1)

  Description : The mesh number for the Gauss-legendre quadrature about β integration (S_y rotation) for the spin quantum-number projection in actual numerical calculation.

- `NSPStot`

  Type : int-type (greater than 0, default value: 0)

  Description : The spin quantum-number.

- `NMPTrans`

  Type : int-type (Positive integer. As a defalut, The number of translational vectors in the sublattice)

  Description : The number of the momentum and lattice translational quantum-number projection. In the case of not to apply the projection, this value must be set as 1.

- `NSROptItrStep`

  Type : int-type (Positive integer, default value: 1000)

  Description : The whole step number to optimize variational parameters by SR method. Only used for `NVMCCalMode`=0.

- `NSROptItrSmp`

  Type : int-type (Positive integer, default value: `NSROptItrStep`/10)

  Description : In the `NSROptItrStep` step, the average values of the each variational parameters at the `NSROptItrStep` step are adopted as the optimized values. Only used for `NVMCCalMode`=0.

- `DSROptRedCut`

  Type : double-type (default value: 0.001)

  Description : The stabilized factor for the SR method by truncation of redundant directions corresponding to $\varepsilon_{\rm wf}$ in the ref. [3].

- `DSROptStaDel`

  Type : double-type (default value: 0.02)

  Description : The stabilized factor for the SR method by modifying diagonal elements in the overwrap matrix corresponding to ε in the ref. [3].

- `DSROptStepDt`

  Type : double-type (default value: 0.02)

  Description : The time step using in the SR method.

- `NVMCWarmUp`

  Type : int-type (Positive integer, default value: 10)

  Description : Idling number for the Malkov chain Montecarlo Methods.

- `NVMCInterval`

  Type : int-type (Positive integer, default value: 1)

  Description : The interval step between samples. The local update will be performed `Nsite` $\times$ `NVMCInterval` times.

- `NVMCSample`

  Type : int-type (Positive integer, default value: 1000)

  Description : The sample numbers to calculate the expected values.

- `NExUpdatePath`

  Type : int-type (Positive integer)

  Description : The option for local update about exchange terms. 0: not update, 1: update. The default value is set as 1 when the local spin exists, otherwise 0.

- `RndSeed`

  Type : int-type (default value: 123456789)

  Description : The initial seed of generating random number. For MPI parallelization, the initial seeds are given by `RndSeed`+my rank+1 at each ranks.

- `NSplitSize`

  Type : int-type (Positive integer, default value: 1)

  Description : The number of processes of MPI parallelization.

- `NStore`

  Type : int-type (Positive integer, default value: 0)

  Description : The option of applying matrix-matrix product to calculate expected values $\langle O_k O_l \rangle$ (0: off, 1: on).

- `ComplexType`

  Type : int-type (`0` or `1`. `0` as a default)

  Description : If it is `0`, only the real part of the variational parameters are optimized. And the real and the imaginary part of them are optimized if this parameter is `1`.

- `OutputMode`

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

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

  In spin system, indices are specified as those on the Bogoliubov representation (See 5.2).

- `CDataFileHead`

  Type : string-type (default : `"zvo"`)

  Description : A header for output files. For example, the output filename for one body Green's function becomes "**xxx_cisajs_yyy.dat**" (xxx are characters set by `CDataFileHead` and yyy are numbers given by numbers from `NDataIdxStart` to `NDataIdxStart` + `NDataQtySmp`).

- `CParaFileHead`

  Type : string-type (default : `"zqp"`)

  Description : A header for output files of the optimized variational parameters. For example, the optimized variational parameters are outputted as **zzz_opt_yyy.dat** (zzz are characters set by `CParaFileHead` and yyy are numbers given by numbers from `NDataIdxStart` to `NDataIdxStart` + `NDataQtySmp`-1).

4.2 Detailed input files

In this section, detailed input files (*.def) are explained. Input files are categorized by the following six parts. The files that are listed in parentheses correspond to the file made by vmcdry.out.

(1) List:
No keyword (namelist.def): This file is a list of input file names with keywords. Each keywords is fixed, but file names are free to be determined.

(2) Basic parameters:
ModPara (modpara.def): Set the parameters for basic parameters such as site number, electron number, Lanczos step *etc.*
LocSpin (locspn.def): Set the location of local spin.

(3) Hamiltonian:
Hamiltonian for mVMC is denoted by

$$\begin{aligned}
\mathcal{H} &= \mathcal{H}_T + \mathcal{H}_U + \mathcal{H}_V + \mathcal{H}_H + \mathcal{H}_E + \mathcal{H}_P + \mathcal{H}_I, && (4.5)\\
\mathcal{H}_T &= -\sum_{i,j}\sum_{\sigma_1,\sigma 2} t_{ij\sigma_1\sigma_2} c_{i\sigma_1}^{\dagger} c_{j\sigma_2}, && (4.6)\\
\mathcal{H}_U &= \sum_i U_i n_{i\uparrow} n_{i\downarrow}, && (4.7)\\
\mathcal{H}_V &= \sum_{i,j} V_{ij} n_i n_j, && (4.8)\\
\mathcal{H}_H &= -\sum_{i,j} J_{ij}^{\mathrm{Hund}} (n_{i\uparrow} n_{j\uparrow} + n_{i\downarrow} n_{j\downarrow}), && (4.9)\\
\mathcal{H}_E &= \sum_{i,j} J_{ij}^{\mathrm{Ex}} (c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{j\downarrow}^{\dagger} c_{i\downarrow} + c_{i\downarrow}^{\dagger} c_{j\downarrow} c_{j\uparrow}^{\dagger} c_{i\uparrow}), && (4.10)\\
\mathcal{H}_P &= \sum_{i,j} J_{ij}^{\mathrm{Pair}} c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{i\downarrow}^{\dagger} c_{j\downarrow}, && (4.11)\\
\mathcal{H}_I &= \sum_{i,j,k,l} \sum_{\sigma_1,\sigma_2,\sigma_3,\sigma_4} I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4}, && (4.12)
\end{aligned}$$

as the format of interactions for electron system. Here, we define the charge density operator with spin σ at site i as $n_{i\sigma} = c_{i\sigma}^{\dagger} c_{i\sigma}$ and the total charge density operator at site i as $n_i = n_{i\uparrow} + n_{i\downarrow}$. Each parameters are specified by the following files, respectively;
Trans (trans.def): $t_{ij\sigma_1\sigma_2}$ in $\mathcal{H}_T$,
CoulombIntra (coulombintra.def): U_i in $\mathcal{H}_U$,
CoulombInter (coulombinter.def): V_{ij} in $\mathcal{H}_V$,
Hund (hund.def): J_{ij}^{Hund} in $\mathcal{H}_H$,
Exchange (exchange.def): J_{ij}^{Ex} in $\mathcal{H}_E$,
PairHop: J_{ij}^{Pair} in $\mathcal{H}_P$,
InterAll: $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}$ in $\mathcal{H}_I$.

(4) Variational parameters to be optimized:
The variational parameters to be optimized are specified by using this categorized

files. In mVMC, the variational wave function is given as

$$\begin{aligned}
|\psi\rangle &= \mathcal{P}_G\mathcal{P}_J\mathcal{P}_{d-h}^{(2)}\mathcal{P}_{d-h}^{(4)}\mathcal{L}^S\mathcal{L}^K\mathcal{L}^P|\phi_{\rm pair}\rangle, &(4.13)\\
\mathcal{P}_G &= \exp\left[\sum_i g_i n_{i\uparrow}n_{i\downarrow}\right], &(4.14)\\
\mathcal{P}_J &= \exp\left[\frac{1}{2}\sum_{i\neq j} v_{ij}(n_i-1)(n_j-1)\right], &(4.15)\\
\mathcal{P}_{d-h}^{(2)} &= \exp\left[\sum_t\sum_{n=0}^{2}(\alpha_{2nt}^d\sum_i\xi_{i2nt}^d+\alpha_{2nt}^h\sum_i\xi_{i2nt}^h)\right], &(4.16)\\
\mathcal{P}_{d-h}^{(4)} &= \exp\left[\sum_t\sum_{n=0}^{4}(\alpha_{4nt}^d\sum_i\xi_{i4nt}^d+\alpha_{4nt}^h\sum_i\xi_{i4nt}^h)\right], &(4.17)\\
\mathcal{L}_S &= \frac{2S+1}{8\pi^2}\int d\Omega P_s(\cos\beta)\hat{R}(\Omega), &(4.18)\\
\mathcal{L}_K &= \frac{1}{N_s}\sum_{\boldsymbol{R}} e^{i\boldsymbol{K}\cdot\boldsymbol{R}}\hat{T}_{\boldsymbol{R}}, &(4.19)\\
\mathcal{L}_P &= \sum_\alpha p_\alpha\hat{G}_\alpha, &(4.20)
\end{aligned}$$

where $\Omega=(\alpha,\beta,\gamma)$ is the Euler angle, $\hat{R}(\Omega)$ is the rotational operator, $P_S(x)$ is the S-th polynomial, $\boldsymbol{K}$ is the momentum operator of the whole system and $\hat{T}_{\boldsymbol{R}}$ is the translational operators corresponding to the translational vector $\boldsymbol{R}$, $\hat{G}_\alpha$ is the point group operator, and p_α is the parity operator, respectively. The details of $\mathcal{P}_{d-h}^{(2)}$ and $\mathcal{P}_{d-h}^{(4)}$ are shown in [3]. The one body part of the wavefunction is represented as the pair function of the real space:

$$|\phi_{\rm pair}\rangle = \left[\sum_{i,j=1}^{N_s} f_{ij}c_{i\uparrow}^\dagger c_{j\downarrow}^\dagger\right]^{N/2}|0\rangle, \qquad (4.21)$$

where N is the number of electrons and N_s is the number of sites. The setting for optimizing variational parameters or not is given by the following files (the parameters for $\mathcal{L}_S$ are specified in the **ModPara** file).

Gutzwiller (gutzwilleridx.def): Set the target parameters g_i in $\mathcal{P}_G$ to be optimized.

Jastrow (jastrowidx.def): Set the target parameters v_{ij} in $\mathcal{P}_J$ to be optimized.

DH2: Set the target 2-site doublon-holon correlation factor $\alpha_{2nt}^{d(h)}$ in $\mathcal{P}_{d-h}^{(2)}$ to be optimized.

DH4: Set the target 4-site doublon-holon correlation factor $\alpha_{4nt}^{d(h)}$ in $\mathcal{P}_{d-h}^{(4)}$ to be optimized.

Orbital (orbitalidx.def): Set the pair orbital f_{ij} in $|\phi_{\rm pair}\rangle$ to be optimized.

TransSym (qptransidx.def): Set the the momentum projection operators $\mathcal{L}_K$ and the lattice translational projection operators $\mathcal{L}_P$.

(5) Initial variational parameters:

Set the initial values of the variational parameters. When the keyword is not setting,

the corresponding parameters are given by random values as default values.
InGutzwiller: Set the initial values of g_i in $\mathcal{P}_G$.
InJastrow: Set the initial values of v_{ij} in $\mathcal{P}_J$.
InDH2: Set the initial values of $\alpha_{2nt}^{d(h)}$ in $\mathcal{P}_{d-h}^{(2)}$.
InDH4: Set the initial values of $\alpha_{4nt}^{d(h)}$ in $\mathcal{P}_{d-h}^{(4)}$.
InOrbital: Set the initial values of f_{ij} in $|\phi_{\rm pair}\rangle$.

(6) Output:
OneBodyG (greenone.def): Set the components of one-body green functions to output.
TwoBodyG (greentwo.def): Set the components of two-body green functions to output.

4.2.1 List file for Input files (namelist.def)

This file determines input filenames correlated with keywords. An example of the file format is shown as follows.

```
ModPara  modpara.def
LocSpin  zlocspn.def
Trans    ztransfer.def
InterAll zinterall.def
Orbital orbitalidx.def
OneBodyG zcisajs.def
TwoBodyG zcisajscktaltdc.def
```

File format

[string01] [string02]

Parameters

- [string01]

  Type : string-type

  Description : Select a word from keywords.

- [string02]

  Type : string-type

  Description : An input filename which is correlated with keywords.

User rules

- After setting keywords at [string 01], half-width state is needed for writing a filename. You can set the filename freely.
- Keywords for input files are shown in Table 4.2.
- Essential keywords are “CalcMod”, “ModPara” , “LocSpin”, “Orbital” and “TransSym”.
- Keywords can be set in random order.
- If keywords or filenames are incorrect, the program is terminated.
- When the head of line is “#”, the line is skipped.

Keywords	Details for corresponding files
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.
Gutzwiller	Gutzwiller factors.
Jastrow	Charge Jastrow factors.
DH2	2-site doublon-holon correlation factors.
DH4	4-site doublon-holon correlation factors.
Orbital*	Pair orbital factors.
TransSym*	Translational and lattice symmetry operation.
InGutzwiller	Initial values of Gutzwiller factors.
InJastrow	Initial values of charge Jastrow factors.
InDH2	Initial values of 2-site doublon-holon correlation factors.
InDH4	Initial values of 4-site doublon-holon correlation factors.
InOrbital	Initial values of pair orbital factors.
OneBodyG	Output components for Green functions $\langle c_{i\sigma}^{\dagger} c_{j\sigma} \rangle$
TwoBodyG	Output components for Correlation functions $\langle c_{i\sigma}^{\dagger} c_{j\sigma} c_{k\tau}^{\dagger} c_{l\tau} \rangle$

Table 4.2: List of the definition files. The files marked * are essential for executing.

4.2.2 ModPara file (modpara.def)

This file determines parameters for calculation. An example of the file format is shown as follows.

```
--------------------
Model_Parameters   0
--------------------
VMC_Cal_Parameters
--------------------
CDataFileHead  zvo
CParaFileHead  zqp
--------------------
NVMCCalMode    0
NLanczosMode   0
--------------------
NDataIdxStart  1
NDataQtySmp    1
--------------------
Nsite          16
Nelectron      8
NSPGaussLeg    1
NSPStot        0
NMPTrans       1
NSROptItrStep  1200
NSROptItrSmp   100
DSROptRedCut   0.001
DSROptStaDel   0.02
DSROptStepDt   0.02
NVMCWarmUp     10
NVMCInterval   1
NVMCSample     1000
NExUpdatePath  0
RndSeed        11272
NSplitSize     1
NStore         1
```

File format

- Lines 1 - 5: Header
- Line 6: [string01] [string02]
- Line 7: [string03] [string04]
- Line 8: Header
- Lines 9 - : [string05] [int01] (or [double01])

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : Set a keyword for header of output files.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : Set a header of output files. For example, the output file of one-body green's functions are named as **xxx_cisajs.dat**, where **xxx** is [string02].

- [string03]

  Type : string-type (blank parameter not allowed)

  Description : Set a keyword for header of output files for variational parameters.

- [string04]

  Type : string-type (blank parameter not allowed)

  Description : Set a header of output files for variational parameters. For example, the output file of optimized variational parameters are named as **xxx_opt.dat**, where **xxx** is [string04].

- [string05]

  Type : string-type

  Description : Select a word from keywords.

- [int01] ([double01])

  Type : int (double)-type (blank parameter not allowed)

  Description : A parameter which is correlated with a keyword.

User rules

- From Line 9: After setting keywords at [string 01], a half-width blank is needed for setting a parameter.
- From Line 9: When the first character of the line is "-", the line is not read and skipped.

Keywords and parameters

- `NVMCCalMode`

  Type : int-type (default value: 0)

  Description : [0] Optimization of variational parameters, [1] Calculation of one body and two body Green's functions.

- `NDataIdxStart`

  Type : int-type (default value: 0)

  Description : An integer for numbering of output files. For `NVMCCalMode`= 0 , `NDataIdxStart` is added at the end of the output files. For `NVMCCalMode` = 1, the files are outputted with the number from `NDataIdxStart` to `NDataIdxStart`+`NDataQtySmp`-1.

- `NDataQtySmp`

  Type : int-type (default value: 1)

  Description : The set number for outputted files (only used for `NVMCCalMode` = 1).

- `Nsite`

  Type : int-type (Positive integer)

  Description : The number of sites.

- `Nelectron`

  Type : int-type (Positive integer)

  Description : The electron number with $\uparrow$ spin. Since the calculation is done at the partial space $S_z = 0$, the electron number with $\uparrow$ spin is equal to that with $\downarrow$ spin.

- `NSPGaussLeg`

  Type : int-type (Positive integer, default value: 8)

  Description : The mesh number for the Gauss-legendre quadrature about β integration (S_y rotation) for the spin quantum-number projection in actual numerical calculation.

- `NSPStot`

  Type : int-type (greater than 0, default value: 0)

  Description : The spin quantum-number.

- `NMPTrans`

  Type : int-type (default value: 1)

  Description : The absolute value gives the number of the momentum and lattice translational quantum-number projection. When the value is negative, the mode of anti-periodic condition turns on. The quantum-number projection is used from the top to `NMPTrans` with the specified weight indicated in `TransSym` file. In the case of not applying the projection, this value must be equal to 1.

- `NSROptItrStep`

  Type : int-type (Positive integer, default value: 1000)

  Description : The whole step number to optimize variational parameters by SR method. Only used for `NVMCCalMode`=0.

- `NSROptItrSmp`

  Type : int-type (Positive integer, default value: `NSROptItrStep`/10)

  Description : In the `NSROptItrStep` step, the average values of the each variational parameters at the `NSROptItrStep` step are adopted as the optimized values. Only used for `NVMCCalMode`=0.

- `DSROptRedCut`

  Type : double-type (default value: 0.001)

  Description : The stabilized factor for the SR method by truncation of redundant directions corresponding to $\varepsilon_{\rm wf}$ in the ref. [3].

- `DSROptStaDel`

  Type : double-type (default value: 0.02)

  Description : The stabilized factor for the SR method by modifying diagonal elements in the overwrap matrix corresponding to ε in the ref. [3].

- `DSROptStepDt`

  Type : double-type

  Description : The time step using in the SR method.

- `NVMCWarmUp`

  Type : int-type (Positive integer, default value: 10)

  Description : Idling number for the Malkov chain Montecarlo Methods.

- `NVMCInterval`

  Type : int-type (Positive integer, default value: 1)

  Description : The interval step between samples. The local update will be performed `Nsite` × `NVMCInterval` times.

- `NVMCSample`

  Type : int-type (Positive integer, default value: 1000)

  Description : The sample numbers to calculate the expected values.

- `NExUpdatePath`

  Type : int-type (Positive integer)

  Description : The option for local update about exchange terms. 0: not update, 1: update for electron system. For Spin system, the value must be 2.

- `RndSeed`

  Type : int-type

  Description : The initial seed of generating random number. For MPI parallelization, the initial seeds are given by `RndSeed`+my rank+1 at each ranks.

- `NSplitSize`

  Type : int-type (Positive integer, default value: 1)

  Description : The number of processes of MPI parallelization.

- `NStore`

  Type : int-type (Positive integer, default value: 0)

  Description : The option of applying matrix-matrix product to calculate expected values $\langle O_k O_l \rangle$ (0: off, 1: on).

4.2.3 LocSpin file (locspn.def)

This file determines sites with localized spins. An example of the file format is shown as follows.

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

File format

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

Parameters

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of localized spins.

- [int02]

  Type : int-type (blank parameter not allowed)

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

- [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer for selecting an electron state whether localized spin or itinerant electron states (0: Itinerant electron state, 1: localized spin state with $S = 1/2$).

Use rules

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

4.2.4 Trans file (trans.def)

The Hamiltonian for general one-body interactions

$$\mathcal{H}_T = -\sum_{ij\sigma_1\sigma 2} t_{ij\sigma_1\sigma 2} c_{i\sigma_1}^{\dagger} c_{j\sigma_2}, \tag{4.22}$$

is added to the whole Hamiltonian by setting the parameters $t_{ij\sigma_1\sigma 2}$. In ver.1.0, σ_1 must be equal to σ_2, since the calculation is only allowed at $S_z = 0$. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of transfer integrals.

- [int02], [int04]

  Type : int-type (blank parameter not allowed)

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

- [int03], [int05]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

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

- [double02]

  Type : double-type (blank parameter not allowed)

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

Use rules

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

4.2.5 InterAll file

The Hamiltonian for general two-body interactions

$$\mathcal{H}_I = \sum_{i,j,k,l} \sum_{\sigma_1,\sigma_2,\sigma_3,\sigma_4} I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} c_{i\sigma_1}^{\dagger} c_{j\sigma_2} c_{k\sigma_3}^{\dagger} c_{l\sigma_4}. \tag{4.23}$$

is added to the whole Hamiltonian by setting the parameters $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4}$. In ver.1.0, this file cannot be used for the spin system. Furthermore, σ_1 and σ_3 must be equal to σ_2, σ_4, respectively, since the calculation is only allowed at $S_z = 0$. An example of file format is shown as follows.

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

File format

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

Parameters

- [string01]

Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

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

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

  Type : int-type (blank parameter not allowed)

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

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

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

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

- [double02]

  Type : double-type (blank parameter not allowed)

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

Use rules

- Headers cannot be omitted.
- Since Hamiltonian must be Hermitian, the following relation must be satisfied, $I_{ijkl\sigma_1\sigma_2\sigma_3\sigma_4} = I^{\dagger}_{lkji\sigma_4\sigma_3\sigma_2\sigma_1}$.
- A program is terminated, when [int01] is different from the total number of generalized two body interactions defined in this file.
- A program is terminated, when [int02]-[int09] are out of range from the defined values.

4.2.6 CoulombIntra file (coulombintra.def)

The Hamiltonian for the coulombintra interactions

$$\mathcal{H}_U = \sum_i U_i n_{i\uparrow} n_{i\downarrow} \tag{4.24}$$

is added to the whole Hamiltonian by setting U_i. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

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

- [int02]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for U_i.

Use rules

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

4.2.7 CoulombInter file (coulombinter.def)

The Hamiltonian for the coulombintrer interactions

$$\mathcal{H}_V = \sum_{i,j} V_{ij} n_i n_j \tag{4.25}$$

is added to the whole Hamiltonian by setting V_{ij}. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

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

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : A value for V_{ij}.

Use rules

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

4.2.8 Hund file (hund.def)

The Hamiltonian for Hund couplings

$$\mathcal{H}_H = -\sum_{i,j} J_{ij}^{\rm Hund}(n_{i\uparrow}n_{j\uparrow} + n_{i\downarrow}n_{j\downarrow}) \tag{4.26}$$

is added to the whole Hamiltonian by setting the parameters $J_{ij}^{\rm Hund}$. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of Hund couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

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

Use rules

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

4.2.9 PairHop file

The Hamiltonian for PairHop couplings

$$\mathcal{H}_P = \sum_{i,j} J_{ij}^{\text{Pair}} c_{i\uparrow}^\dagger c_{j\uparrow} c_{i\downarrow}^\dagger c_{j\downarrow} \tag{4.27}$$

is added to the whole Hamiltonian by setting the parameters J_{ij}^{Pair}. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of PairHop couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

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

Use rules

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

4.2.10 Exchange file (exchange.def)

The Hamiltonian for exchange couplings

$$\mathcal{H}_E = \sum_{i,j} J_{ij}^{\rm Ex} (c_{i\uparrow}^{\dagger} c_{j\uparrow} c_{j\downarrow}^{\dagger} c_{i\downarrow} + c_{i\downarrow}^{\dagger} c_{j\downarrow} c_{j\uparrow}^{\dagger} c_{i\uparrow}) \tag{4.28}$$

is added to the whole Hamiltonian by setting $J_{ij}^{\rm Ex}$. An example of the file format is shown as follows.

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of Exchange couplings.

- [int02], [int03]

  Type : int-type (blank parameter not allowed)

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

- [double01]

  Type : double-type (blank parameter not allowed)

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

Use rules

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

4.2.11 Gutzwiller file (gutzwiller.def)

This file sets the calculation conditions of Gutzwiller factors

$$\mathcal{P}_G = \exp\left[\sum_i g_i n_{i\uparrow} n_{i\downarrow}\right]. \tag{4.29}$$

A site number i and the variational parameters g_i are specified. An example of the file format is shown as follows.

```
======================
NGutzwillerIdx 2
ComplexType 0
======================
======================
   0     0
   1     0
   2     0
   3     1
(continue...)
  12     1
  13     0
  14     0
  15     0
   0     1
   1     0
```

File format

In the following, we define the whole number of sites as N_s and variational parameters as N_g, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Line 3: [string02] [int02]
- Lines 4 - 5: Header
- Lines 6 - (5+N_s): [int03] [int04]
- Lines (6+N_s) - (5+N_s+N_g): [int05] [int06]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of variational parameters g_i. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters g_i.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for indicating the double or complex type of variational parameters g_i. You can freely give a name of the keyword.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer indicates the double or complex type of variational parameters g_i (0: double, 1: complex).

- [int03]

  Type : int-type (blank parameter not allowed)

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

- [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters g_i (0 <= [int04] < [int01]).

- [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of variational parameters (0 <= [int05] < [int01]).

- [int06]

  Type : int-type (blank parameter not allowed)

  Description : An integer to select the target of variational parameters indicated at [int05] to be optimized or not (0: not optimize, 1: optimize).

User rules

- Headers cannot be omitted.
- A program is terminated, when components of variational parameters are double counted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.
- A program is terminated, when [int02] - [int06] are out of range from the defined values.

4.2.12 Jastrow file (jastrow.def)

This file sets the calculation conditions of Jastrow factors

$$\mathcal{P}_J = \exp\left[\frac{1}{2}\sum_{i\neq j} v_{ij}(n_i-1)(n_j-1)\right] \tag{4.30}$$

Site numbers i j, and the variational parameters v_{ij} are specified. An example of the file format is shown as follows.

```
======================
NJastrowIdx 5
ComplexType 0
======================
======================
   0     1     0
   0     2     1
   0     3     0
 (continue...)
   0    1
   1    1
   2    1
   3    1
   4    1
```

File format

In the following, we define the total number of sites as N_s and variational parameters as N_j, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Line 3: [string02] [int02]
- Lines 4 - 5: Header
- Lines 6 - (5+$N_s \times (N_s-1)$): [int03] [int04] [int05]
- Lines (6+$N_s \times (N_s-1)$) - (5+$N_s \times (N_s-1)$+N_j): [int06] [int07]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for total number of variational parameters v_{ij}. You can freely give a name of the keyword.

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters v_{ij}.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for indicating the double or complex type of variational parameters v_{ij}. You can freely give a name of the keyword.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer indicates the double or complex type of variational parameters v_{ij} (0: double, 1: complex).

- [int03], [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int03], [int04] < `Nsite`).

- [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters v_{ij} (0 <= [int05] < [int01]).

- [int06]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of variational parameters (0 <= [int06] < [int01]).

- [int07]

  Type : int-type (blank parameter not allowed)

  Description : An integer to select the target of variational parameters indicated at [int06] to be optimized or not (0: not optimize, 1: optimize).

User rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.
- A program is terminated, when [int02] - [int07] are out of range from the defined values.

4.2.13 DH2 file

This file sets the calculation conditions of 2-site doublon-holon correlation factors

$$\mathcal{P}_{d-h}^{(2)} = \exp\left[\sum_t \sum_{n=0}^{2} (\alpha_{2nt}^d \sum_i \xi_{i2nt}^d + \alpha_{2nt}^h \sum_i \xi_{i2nt}^h)\right]. \tag{4.31}$$

A site number i, the two sites around i site and the variational parameters $\alpha_{2nt}^{d(h)}$ which have t kinds at each sites are specified. The details of the parameters $\alpha_{2nt}^{d(h)}$ and the operator $\xi_{i2nt}^{d(h)}$ are shown in ref. [3]. An example of the file format is shown as follows.

```
====================================
NDoublonHolon2siteIdx 2
ComplexType 0
====================================
====================================
   0     5   15    0
   0    13    7    1
 (continue...)
  15     8    2    1
   0     1
(continue...)
  11     1
```

File format

In the following, we define the total number of sites as N_s and variational parameters as $N_{\rm dh2}$, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Line 3: [string02] [int02]
- Lines 4 - 5: Header
- Lines 6 - (5+$N_s \times N_{\rm dh2}$): [int03] [int04] [int05] [int06]
- Lines (6+$N_s \times N_{\rm dh2}$) - (5+$(N_s + 6) \times N_{\rm dh2}$): [int07] [int08]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for indicating the double or complex type of variational parameters. You can freely give a name of the keyword.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer indicates the double or complex type of variational parameters (0: double, 1: complex).

- [int03], [int04], [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int03], [int04], [int05] < `Nsite`).

- [int06]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters (0 <= [int06] < [int01]).

- [int07]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of variational parameters. The value is $(2n+s)\times$[int01]$+t$, where n, s and t are given by the following relation:

  - n: The number of doublon (holon) around the center site (0, 1, 2),

  - s: When the center is doublon (holon), s=0 (1),

  - t: The kind of variational parameters ($0, \cdots$ [int1]-1).

- [int08]

  Type : int-type (blank parameter not allowed)

  Description : An integer to select the target of variational parameters indicated at [int07] to be optimized or not (0: not optimize, 1: optimize).

User rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.
- A program is terminated, when [int02] - [int08] are out of range from the defined values.

4.2.14 DH4 file

This file sets the calculation conditions of 4-site doublon-holon correlation factors

$$\mathcal{P}_{d-h}^{(4)} = \exp\left[\sum_t \sum_{n=0}^{4} (\alpha_{4nt}^d \sum_i \xi_{i4nt}^d + \alpha_{4nt}^h \sum_i \xi_{i4nt}^h)\right] \tag{4.32}$$

A site number i, the four sites around i site and the variational parameters $\alpha_{4nt}^{d(h)}$ which have t kinds at each sites are specified. The details of the parameters $\alpha_{4nt}^{d(h)}$ and the operator $\xi_{i4nt}^{d(h)}$ are shown in ref. [3]. An example of the file format is shown as follows.

```
====================================
NDoublonHolon4siteIdx 1
ComplexType 0
====================================
====================================
   0     1    3    4   12    0
   1     2    0    5   13    0
 (continue...)
  15    12   14    3   11    0
   0     1
(continue...)
   9     1
```

File format

In the following, we define the total number of sites as N_s and variational parameters as $N_{\rm dh4}$, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Line 3: [string02] [int02]
- Lines 4 - 5: Header
- Lines 6 - (5+$N_s \times N_{\rm dh4}$): [int03] [int04] [int05] [int06] [int07] [int08]
- Lines (6+$N_s \times N_{\rm dh4}$) - (5+$(N_s + 10) \times N_{\rm dh4}$): [int09] [int10]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for indicating the double or complex type of variational parameters. You can freely give a name of the keyword.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer indicates the double or complex type of variational parameters (0: double, 1: complex).

- [int03], [int04], [int05], [int06], [int07]

  Type : int-type (blank parameter not allowed)

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

- [int08]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters (0 <= [int08] < [int01]).

- [int09]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of variational parameters. The value is $(2n+s)\times$[int01]$+t$, where n, s and t are given by the following relation:

  - n: The number of doublon (holon) around the center site (0, 1, 2, 3, 4),

  - s: When the center is doublon (holon), s=0 (1),

  - t: The kind of variational parameters (0, $\cdots$ [int1]-1).

- [int10]

  Type : int-type (blank parameter not allowed)

  Description : An integer to select the target of variational parameters indicated at [int09] to be optimized or not (0: not optimize, 1: optimize).

User rules

- Headers cannot be omitted.
- A program is terminated, when components of variational parameters are double counted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.
- A program is terminated, when [int02] - [int10] are out of range from the defined values.

4.2.15 Orbital file (orbitalidx.def)

This file sets the calculation conditions of pair orbitals

$$|\phi_{\rm pair}\rangle = \left[\sum_{i,j=1}^{N_s} f_{ij} c_{i\uparrow}^{\dagger} c_{j\downarrow}^{\dagger}\right]^{N/2} |0\rangle. \tag{4.33}$$

Site numbers i, j and the variational parameters f_{ij} are indicated. An example of the file format is shown as follows.

```
====================================
NOrbitalIdx 64
ComplexType 0
====================================
====================================
   0     0     0
   0     1     1
   0     2     2
   0     3     3
 (continue...)
  15     9    62
  15    10    63
   0    1
   1    1
(continue...)
  62    1
  63    1
```

File format

In the following, we define the total number of sites as N_s and variational parameters as $N_{\rm o}$, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Line 3: [string02] [int02]
- Lines 4 - 5: Header
- Lines 6 - (5+N_s^2): [int03] [int04] [int05] [int06]
- Lines (6+N_s^2)- (5+$N_s^2 + N_{\rm o}$): [int06] [int07]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters.

- [string02]

  Type : string-type (blank parameter not allowed)

  Description : A keyword for indicating the double or complex type of variational parameters. You can freely give a name of the keyword.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer indicates the double or complex type of variational parameters (0: double, 1: complex).

- [int03], [int04]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int03], [int04] < `Nsite`).

- [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters (0 <= [int05] < [int01]).

- [int06]

  Type : int-type

  Description : When the mode of the anti-periodic condition turns on (the mode turns on when the value of `NMPTrans` in `ModPara` file is negative), the sign of f_{ij} is specified by setting [int06] = ±1. This term can be omitted when the mode of the anti-periodic condition is off.

- [int07]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of variational parameters (0 <= [int06] < [int01]).

- [int08]

  Type : int-type (blank parameter not allowed)

  Description : An integer to select the target of variational parameters indicated at [int06] to be optimized or not (0: not optimize, 1: optimize).

User rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.

- A program is terminated, when [int02] - [int09] are out of range from the defined values.

4.2.16 TransSym file (qptransidx.def)

This file sets the weight and corresponding site numbers of momentum projection $\mathcal{L}_K = \frac{1}{N_s}\sum_{\boldsymbol{R}} e^{i\boldsymbol{K}\cdot\boldsymbol{R}}\hat{T}_{\boldsymbol{R}}$ and lattice translational projection $\mathcal{L}_P = \sum_\alpha p_\alpha \hat{G}_\alpha$. The patterns of projection are indicated by $(\alpha, \boldsymbol{R})$. We note that the weight must be equal to 1.0 when the projection is not done. An example of the file format is shown as follows.

```
====================================
NQPTrans 4
====================================
== TrIdx_TrWeight_and_TrIdx_i_xi  ==
====================================
   0  1.000000
   1  1.000000
   2  1.000000
   3  1.000000
   0     0    0
 (continue...)
   3    12    1
   3    13    2
```

File format

In the following, we define the total number of sites as N_s and projection patterns as N_{TS}, respectively.

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3 - 5: Header
- Lines 6 - (5+N_{TS}): [int02] [double01]
- Lines (6+N_{TS}) - (5+$(N_s + 1) \times N_{\mathrm{TS}}$): [int03] [int04] [int05] [int06]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of projection patterns.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving the projection pattern $(\alpha, \boldsymbol{R})$ (0 <= [int02] < [int01]).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : The weight $p_\alpha \cos(\boldsymbol{K} \cdot \boldsymbol{R})$ of the projection pattern $(\alpha, \boldsymbol{R})$.

- [int03]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving kinds of the projection pattern $(\alpha, \boldsymbol{R})$ (0 <= [int03] < [int01]).

- [int04], [int05]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving a site index (0 <= [int04], [int05] < `Nsite`). The site number [int05] is given by applying the translation and point group transformation indicated by [int03] to the site [int04].

- [int06]

  Type : int-type

  Description : When the mode of the anti-periodic condition turns on (the mode turns on when the value of `NMPTrans` in `ModPara` file is negative), the sign of the translational operator is specified by setting [int06] = ± 1. This term can be omitted when the mode of the anti-periodic condition is off.

User rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of projection patterns defined in this file.
- A program is terminated, when [int02] - [int06] are out of range from the defined values.

4.2.17 Files to set initial values of variational parameters

This file sets the initial values of variational parameters. The kinds of variational parameters are specified by setting the following keywords in `List` file (namelist.def): `InGutzwiller`, `InJastrow`, `InDH2`, `InDH4`, `InOrbital`.
The file format is common and an example of the `InJastrow` file is shown as follows.

```
=====================
NJastrowIdx  28
=====================
== i_j_JastrowIdx  ===
=====================
0 -8.909963465082626488e-02  0.000000000000000000e+00
1  5.521681211878626955e-02  0.000000000000000000e+00
(continue...)
27 -9.017586139930480749e-02  0.000000000000000000e+00
```

File format

In the following, we define the total number of variational parameters as N_v.

- Line 1: Header
- Line 2: [string01] [int01]
- Lines 3 - 5: Header
- Lines 6 - (5+N_v): [int03] [double01] [double02]

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

  Description : An integer giving total number of variational parameters.

- [int02]

  Type : int-type (blank parameter not allowed)

  Description : An integer setting kinds of variational parameters (0 <= [int02] < [int01]).

- [double01]

  Type : double-type (blank parameter not allowed)

  Description : The real part of the variational parameter indicated by [int01].

- [double02]

  Type : double-type

  Description : The imaginary part of the variational parameter indicated by [int01].

User rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of variational parameters defined in this file.

4.2.18 OneBodyG file (greenone.def)

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

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

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

- [int02], [int04]

  Type : int-type (blank parameter not allowed)

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

- [int03], [int05]

  Type : int-type (blank parameter not allowed)

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

Use rules

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

4.2.19 TwoBodyG file (greentwo.def)

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

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

File format

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

Parameters

- [string01]

  Type : string-type (blank parameter not allowed)

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

- [int01]

  Type : int-type (blank parameter not allowed)

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

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

  Type : int-type (blank parameter not allowed)

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

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

  Type : int-type (blank parameter not allowed)

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

Use rules

- Headers cannot be omitted.
- A program is terminated, when [int01] is different from the total number of two-body Green's functions defined in this file.
- A program is terminated, when [int02]-[int09] are out of range from the defined values.

4.3 Output files

The list of output files are shown in Table 4.3, where *** and xxx are the header indicated by `CParaFileHead`,`CDataFileHead` in `ModPara` file, respectively. yyy is a number given by `NDataIdxStart`$\cdots$`NDataIdxStart`+`NDataQtySmp`, where both `NDataIdxStart` and `NDataQtySmp` are defined in `ModPara` file and zzz is a number given by `NDataIdxStart` in `ModPara`.

Name	Details for corresponding files
***_opt.dat	All optimized parameters.
***_gutzwiller_opt.dat	Optimized gutzwiller factors.
***_jastrow_opt.dat	Optimized jastrow factors.
***_doublonHolon2site_opt.dat	Optimized 2-site doublon-holon correlation factors.
***_doublonHolon4site_opt.dat	Optimized 4-site doublon-holon correlation factors.
***_orbital_opt.dat	Optimized pair orbital factors.
xxx_out_yyy.dat	Energy and deviation.
xxx_var_yyy.dat	Progress information for optimizing variational parameters.
xxx_CalcTimer.dat	Computation time for each processes.
xxx_time_zzz.dat	Progress information for MonteCalro samplings.
xxx_cisajs_yyy.dat	One body Green's functions.
xxx_cisajscktalt_yyy.dat	Correlation functions.

Table 4.3: List of the output files.

4.3.1 Output file for variational parameters (***_opt.dat)

The average and deviation values of variational parameters and the energy optimized by the SR method are outputted in the following order:

$$\langle H\rangle, \langle H^2\rangle, g_i, v_{ij}, \alpha_{2nt}^{d(h)}, \alpha_{4nt}^{d(h)}, f_{ij}.$$

The type of average values is a complex number, while that of the deviation is a real number. Since the initial values of all variational parameters are specified at the beginning of the calculation, the calculation of physical quantities is done by using this file file. Here, *** is the header indicated by `CParaFileHead` in `ModPara` file.

4.3.2 Output files for variational parameters at each steps (xxx_var_yyy.dat)

The average and deviation values of variational parameters and the energy optimized by the SR method are postscripted by the order same as $$$_opt.dat file (the deviation is always outputted as 0). Here, xxx is the header indicated by `CDataFileHead` in `ModPara` file and yyy is a number given by `NDataIdxStart`$\cdots$`NDataIdxStart`+`NDataQtySmp`, where both `NDataIdxStart` and `NDataQtySmp` are defined in `ModPara` file.

4.3.3 Output file for gutzwiller factors (***_gutzwiller_opt.dat)

The optimized Gutzwiller factors by SR method are outputted. The file format is same as the `InGutzwiller` file defined in Sec. 4.2.17.

4.3.4 Output file for jastrow factors (***_jastrow_opt.dat)

The optimized Jastrow factors by SR method are outputted. The file format is same as the `InJastrow` file defined in Sec. 4.2.17.

4.3.5 Output file for doublonHolon 2-site factors (***_doublonHolon2site_opt.dat)

The optimized 2-site doublon-holon crrelation factors by SR method are outputted. The file format is same as the `InDH2` file defined in Sec. 4.2.17.

4.3.6 Output file for doublonHolon 4-site factors (***_doublonHolon4site_opt.dat)

The optimized 4-site doublon-holon crrelation factors by SR method are outputted. The file format is same as the `InDH4` file defined in Sec. 4.2.17.

4.3.7 Output file for pair orbitals (***_orbital_opt.dat)

The optimized pair orbitals by SR method are outputted. The file format is same as the `InOrbital` file defined in Sec. 4.2.17.

4.3.8 xxx_out_yyy.dat

The calculation information at each bins are outputted in the order:

$$\langle H\rangle, \langle H^2\rangle, \frac{\langle H^2\rangle - \langle H\rangle^2}{\langle H\rangle^2}.$$

The type of $\langle H\rangle$ is a complex number, and that of the others is a real number. Here, xxx is the header indicated by `CDataFileHead` in `ModPara` file and yyy is a number given by `NDataIdxStart`$\cdots$`NDataIdxStart`+`NDataQtySmp`, where both `NDataIdxStart` and `NDataQtySmp` are defined in `ModPara` file. An example of outputted file is shown as follows.

```
1.151983765704212992e+01  8.124622418360909482e-01  \
1.619082955438887268e+02  2.019905203939084959e-01
1.288482613817423150e+01  5.006903733262847433e-01  \
1.972000325276957824e+02  1.824505193695792893e-01
1.308897206011880421e+01  5.701244886956570168e-01  \
2.072610167083121837e+02  2.029162857569105916e-01
...
```

4.3.9 xxx_CalcTimer.dat

After finishing calculation, the processing time is outputted in the order of the name, the number assigned by the process and the seconds at each processes. An example of outputted file is shown as follows.

```
All                        [0]     15.90724
Initialization             [1]      0.04357
  read options            [10]      0.00012
  ReadDefFile             [11]      0.00082
  SetMemory               [12]      0.00002
  InitParameter           [13]      0.03026
VMCParaOpt                 [2]     15.86367
  VMCMakeSample            [3]     12.85650
...
```

4.3.10 xxx_time_zzz.dat

The calculation information at each bins are outputted in the order of the sampling number, the acceptance ratio for hopping and exchange term (acc_hopp, acc_ex), trial numbers to update for hopping and exchange term (n_hopp, n_ex) and the time stamp. Here, xxx is the header indicated by `CDataFileHead` in `ModPara` file and zzz is a number given by `NDataIdxStart` in `ModPara`. An example of outputted file is shown as follows.

```
00000  acc_hop acc_ex  n_hop    n_ex     : Mon Jul 25 14:03:29 2016
00001  0.59688 0.00000 320      0        : Mon Jul 25 14:03:30 2016
00002  0.47727 0.00000 176      0        : Mon Jul 25 14:03:30 2016
00003  0.50000 0.00000 176      0        : Mon Jul 25 14:03:30 2016
00004  0.49432 0.00000 176      0        : Mon Jul 25 14:03:30 2016
00005  0.57386 0.00000 176      0        : Mon Jul 25 14:03:30 2016
00006  0.55114 0.00000 176      0        : Mon Jul 25 14:03:30 2016
...
```

4.3.11 xxx_cisajs_yyy.dat

The values of one body Green's functions are outputted in the order indicated by `OneBodyG` file. Here, xxx is the header indicated by `CDataFileHead` in `ModPara` file and yyy is a number given by `NDataIdxStart`$\cdots$`NDataIdxStart`+`NDataQtySmp`, where both `NDataIdxStart` and `NDataQtySmp` are defined in `ModPara` file.

4.3.12 xxx_cisajscktalt_yyy.dat

The values of two body Green's functions are outputted in the order indicated by `OneBodyG` file. Here, xxx is the header indicated by `CDataFileHead` in `ModPara` file and yyy is a number given by `NDataIdxStart`$\cdots$`NDataIdxStart`+`NDataQtySmp`, where both `NDataIdxStart` and `NDataQtySmp` are defined in `ModPara` file.

5
Algorithm

5.1 Variational MonteCalro Method

In Variational MonteCalro (VMC) method, the importance sampling is performed in the system constructed by the Malkov chain toward the appropriate complete system. Here, we choose the real spatial arrangement $\{|x\rangle\}$ at $S_z = 0$ as the complete system.

$$|x\rangle = \prod_{n=1}^{N/2} c^\dagger_{r_{n\uparrow}} \prod_{n=1}^{N/2} c^\dagger_{r_{n\downarrow}} |0\rangle, \tag{5.1}$$

where we set $r_{n\sigma}$ as the position of n-th electron with $\sigma(=\uparrow$ or $\downarrow)$ spin.

5.1.1 Importance sampling

When the importance of Malkov chain is defined by

$$\rho(x) = \frac{|\langle x|\psi\rangle|^2}{\langle\psi|\psi\rangle} \geq 0, \sum x\rho(x) = 1, \tag{5.2}$$

the expected value of the operator A is given by

$$\langle A\rangle = \frac{\langle\psi|A|\psi\rangle}{\langle\psi|\psi\rangle} = \sum_x \frac{\langle\psi|A|x\rangle\langle x|\psi\rangle}{\langle\psi|\psi\rangle} = \sum_x \rho(x)\frac{\langle\psi|A|x\rangle}{\langle\psi|x\rangle}. \tag{5.3}$$

In VMC, the summation in terms of x is replaced by the importance sampling. Then, local Green's function $G_{ij\sigma\sigma'}(x)$ is defined by

$$G_{ij\sigma\sigma'}(x) = \frac{\langle\psi|c^\dagger_{i\sigma}c_{j\sigma'}|\psi\rangle}{\langle\psi|x\rangle}. \tag{5.4}$$

In mVMC, the Mersenne twister method is used for sampling as the random number generator [8].

5.2 Bogoliubov representation

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

$$S_{iz} = \sum_{\sigma=-S}^{S} \sigma c_{i\sigma}^{\dagger} c_{i\sigma} \tag{5.5}$$

$$S_i^+ = \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1)-\sigma(\sigma+1)} c_{i\sigma+1}^{\dagger} c_{i\sigma} \tag{5.6}$$

$$S_i^- = \sum_{\sigma=-S}^{S-1} \sqrt{S(S+1)-\sigma(\sigma+1)} c_{i\sigma}^{\dagger} c_{i\sigma+1} \tag{5.7}$$

5.3 Relation between Pfaffian Slater determinant and single Slater determinant

In this section, we show relation between Pfaffian Slater determinant and single Slater determinant. We also discuss meaning of the singular value decomposition of coefficients f_{ij}.

5.3.1 Relation between f_{ij} and $\Phi_{in\sigma}$

Pfaffian Slater determinant [one-body part of the many-variable variational Monte Carlo (mVMC) method] is defined as

$$|\phi_{\rm Pf}\rangle = \Big(\sum_{i,j=1}^{N_s} f_{ij} c_{i\uparrow}^{\dagger} c_{j\downarrow}^{\dagger} \Big)^{N_e/2} |0\rangle, \tag{5.8}$$

where N_s is number of sites, N_e is number of total particles, and f_{ij} are variational parameters. For simplicity, we assume that f_{ij} are real number. Single Slater determinant is defined as

$$|\phi_{\rm SL}\rangle = \Big(\prod_{n=1}^{N_e/2} \psi_{n\uparrow}^{\dagger} \Big) \Big(\prod_{m=1}^{N_e/2} \psi_{m\downarrow}^{\dagger} \Big) |0\rangle, \tag{5.9}$$

$$\psi_{n\sigma}^{\dagger} = \sum_{i=1}^{N_s} \Phi_{in\sigma} c_{i\sigma}^{\dagger}. \tag{5.10}$$

We note that Φ is the normalized orthogonal basis, i.e,

$$\sum_{i=1}^{N_s} \Phi_{in\sigma} \Phi_{im\sigma} = \delta_{nm}, \tag{5.11}$$

where δ_{nm} is the Kronecker's delta. Due to this normalized orthogonality, we obtain following relation:

$$[\psi_{n\sigma}^{\dagger}, \psi_{m\sigma}]_+ = \delta_{nm}, \tag{5.12}$$

$$G_{ij\sigma} = \langle c_{i\sigma}^{\dagger} c_{j\sigma} \rangle = \frac{\langle \phi_{\rm SL} | c_{i\sigma}^{\dagger} c_{j\sigma} | \phi_{\rm SL} \rangle}{\langle \phi_{\rm SL} | \phi_{\rm SL} \rangle} \tag{5.13}$$

$$= \sum_n \Phi_{in\sigma} \Phi_{jn\sigma}. \tag{5.14}$$

Here, we rewrite ϕ_{SL} and obtain explicit relation between f_{ij} and $\Phi_{in\sigma}$. By using the commutation relation for $\psi^{\dagger}_{n\sigma}$, we rewrite ϕ_{SL} as

$$|\phi_{\mathrm{SL}}\rangle \propto \prod_{n=1}^{N_e/2} \Big(\psi^{\dagger}_{n\uparrow}\psi^{\dagger}_{\mu(n)\downarrow}\Big)|0\rangle, \tag{5.15}$$

where $\mu(n)$ represents permutation of sequence of $n = 1, 2, \cdots, N_e/2$. For simplicity, we take identity permutation and obtain the relation

$$|\phi_{\mathrm{SL}}\rangle \propto \prod_{n=1}^{N_e/2} \Big(\psi^{\dagger}_{n\uparrow}\psi^{\dagger}_{n\downarrow}\Big)|0\rangle = \prod_{n=1}^{N_e/2} K^{\dagger}_{n}|0\rangle \tag{5.16}$$

$$\propto \Big(\sum_{n=1}^{\frac{N_e}{2}} K^{\dagger}_{n}\Big)^{\frac{N_e}{2}}|0\rangle = \Big(\sum_{i,j=1}^{N_s}\Big[\sum_{n=1}^{\frac{N_e}{2}} \Phi_{in\uparrow}\Phi_{jn\downarrow}\Big]c^{\dagger}_{i\uparrow}c^{\dagger}_{j\downarrow}\Big)|0\rangle, \tag{5.17}$$

where $K^{\dagger}_{n} = \psi^{\dagger}_{n\uparrow}\psi^{\dagger}_{n\downarrow}$ and we use the relation $K^{\dagger}_{n}K^{\dagger}_{m} = K^{\dagger}_{m}K^{\dagger}_{n}$. This result shows that f_{ij} can be expressed by the coefficients of the single Slater determinant as

$$f_{ij} = \sum_{n=1}^{\frac{N_e}{2}} \Phi_{in\uparrow}\Phi_{jn\downarrow}. \tag{5.18}$$

We note that this is one of expression of f_{ij} for single Slater determinant, i.e, f_{ij} depend on the pairing degrees of freedom (choices of $\mu(n)$) and gauge degrees of freedom (we can arbitrary change the sign of Φ as $\Phi_{in\sigma} \rightarrow -\Phi_{in\sigma}$). This large degrees of freedom is the origin of huge redundancy of f_{ij}.

5.3.2 Singular value decomposition of f_{ij}

We define matrices F, $\Phi_{\uparrow}$, $\Phi_{\downarrow}$, and Σ as

$$(F)_{ij} = f_{ij}, \quad (\Phi_{\uparrow})_{in} = \Phi_{in\uparrow}, \quad (\Phi_{\downarrow})_{in} = \Phi_{in\downarrow}, \tag{5.19}$$

$$\Sigma = \mathrm{diag}[1, \cdots, 1, 0, 0, 0] \quad (\#\text{ of } 1 = N_e/2). \tag{5.20}$$

By using these notations, we can describe the singular value decomposition of f_{ij} (or equivalently F) as

$$F = \Phi_{\uparrow}\Sigma\Phi^{t}_{\downarrow}. \tag{5.21}$$

This result indicates that f_{ij} can be described by the mean-field solutions if the number of nonzero singular values are $N_e/2$ and all the nonzero singular values of F are one. In other word, the singular values including their numbers offers the quantitative criterion how the Pfaffian Slater determinant deviates from the single Slate determinant.

6
Acknowledgement

T.B.D

A
Program for the unrestricted Hartree-Fock approximation

In mVMC package, there is a program to calculate the initial values of the pair orbital parameters f_{ij} by using the unrestricted Hartree-Fock (UHF) approximation (relation between Pfaffian Slater determinant and single Slater determinant is explained in sec. 5.3). It is noted that the target system of this program is the itinerant electron system.

A.1 Overview

In UHF approximation, two-body interaction terms are approximated as one-body interaction terms by taking into account of the fluctuation, $\delta A \equiv A - \langle A \rangle$, up to the first order. As an example, we consider the inter-site coulomb interactions

$$\mathcal{H}_V = \sum_{i,j} V_{ij} n_i n_j, \tag{A.1}$$

where we define $i \equiv (i, \sigma)$, $j \equiv (j, \sigma')$ for simplicity. Then, the interaction terms can be approximated as

$$\begin{aligned} n_i n_j &= (\langle n_i \rangle + \delta n_i)(\langle n_j \rangle + \delta n_j) - \left[\langle c_i^\dagger c_j \rangle + \delta (c_i^\dagger c_j) \right] \left[\langle c_j^\dagger c_i \rangle + \delta (c_j^\dagger c_i) \right] \\ &\sim \langle n_i \rangle n_j + \langle n_j \rangle n_i - \langle c_i^\dagger c_j \rangle c_j^\dagger c_i - \langle c_j^\dagger c_i \rangle c_i^\dagger c_j - \langle n_i \rangle \langle n_j \rangle + \langle c_j^\dagger c_i \rangle \langle c_i^\dagger c_j \rangle. \end{aligned} \tag{A.2}$$

Also for other types of interaction, the problem can be attributed to a one-body problem by using a similar approximation. Actual calculation is performed iteratively until that self-consistent solution for the mean values of the above observables are obtained.

A.1.1 Source code

A set of source codes are included in the directory `src/ComplexUHF/src`.

A.1.2 How to compile

To compile source codes, move to the directory just below the main directory of mVMC, and execute

```
$ make mvmc
```

in a similar way as the compile of mVMC. After compiling, an executable file `UHF` is generated in `src/ComplexUHF/src`.

A.1.3 Input files

A file for assigning input files (namelsit.def)

The following files are needed to use the program of UHF. The format of `namelist.def` is the same as defined in 4.2.1.

- `ModPara`
- `LocSpin`
- `Trans`
- `CoulombIntra`
- `CoulombInter`
- `Hund`
- `PairHop`
- `Exchange`
- `Orbital`
- `Initial`

Although the format of these files are the same as those for mVMC basically, the following items are different:

- Parameters assigned in `ModPara` file.
- Addition of `Initial` file.

We explain details of the format of these files as follows.

Parameters assigned in ModPara file

The parameters needed in the program of UHF are as follows:

- `Nsite`
- `Ne`
- `Mix`
- `EPS`
- `IterationMax`

The parameters, `Nsite` and `Ne`, are common as mVMC. The other three parameters are specific to UHF:

- `Mix`
  Linear mixing is assigned by double-type. When mix=1, a new Green's function is fully updated without using a old one.

- `EPS`
  A condition for convergence is assigned by int-type. When a residual error between a new Green's function and a previous one is less than $10^{-\texttt{eps}}$, the iteration of calculation is stopped.

- `IterationMax`
  A maximum number of the loop is assigned by int-type.

If there are the other parameters for mVMC in this file , warning is output to the standard output (the calculation is not stopped).

Initial file

Initial values of Green's function $G_{ij\sigma_1\sigma_2} \equiv \langle c^{\dagger}_{i\sigma_1} c_{j\sigma_2} \rangle$ are given. The format is the same as `Trans` file, and instead of $t_{ij\sigma_1\sigma_2}$, values of $G_{ij\sigma_1\sigma_2}$ are described. Green's function is set as zero if values are not given.

A.2 Usage

Calculation of UHF is performed by the same way as mVMC, i.e., by executing the command

```
$ UHF namelist.def
```

The routine of the calculation is as follows.

1. Reading files
2. Construction of a Hamiltonian
3. Self-consistent calculation of Green's function
4. Output of f_{ij} and other files

Examples of output after calculation are as follows.

- zvo_result.dat: The energy and the particle number are output.

```
energy -15.2265348135
num    36.0000000000
```

- zvo_check.dat: The step number of the iteration, the mean of the absolute value of the residual error in Green's function, the energy in convergence process, and the particle number are output in order.

```
 0  0.004925645652 -544.963484605164 36.000000
 1  0.002481594941 -278.304285708488 36.000000
 2  0.001274395448 -147.247026925130 36.000000
 3  0.000681060599 -82.973664527606 36.000000
...
```

- zvo_UHF_cisajs.dat: Convergent one-body Green's function $G_{ij\sigma_1\sigma_2} \equiv \langle c_{i\sigma_1}^{\dagger} c_{j\sigma_2} \rangle$ is output.
  For all the components, $i, \sigma_1, j, \sigma_2, \mathrm{Re}\,[G_{ij\sigma_1\sigma_2}]\,, \mathrm{Im}\,[G_{ij\sigma_1\sigma_2}]$ are output in order.

```
     0    0    0    0 0.5037555283 0.0000000000
     0    0    0    1 0.4610257618 0.0003115503
     0    1    0    0 0.4610257618 -0.0003115503
     0    1    0    1 0.4962444717 0.0000000000
  ...
```

- zvo_eigen.dat: Convergent eigenvalues of the Hamiltonian are output in ascending order.

```
  1  -2.9425069199
  2  -2.9425069198
  3  -1.5005359205
  ...
```

- zvo_gap.dat: For the total electron number N_{tot}, the energy difference $\Delta E = E(N_{\mathrm{tot}} + 1) - E(N_{\mathrm{tot}})$ is output.

```
   5.2208232631
```

- zvo_orbital_opt.dat: f_{ij} generated from the Slater determinant. The file with the same format as `InOrbital` file is output. By referring `Orbital` file, f_{ij} is calculated (for the same type of parameters, the averaged value is calculated).

References

[1] M. Imada, T. Miyake, J. Phys. Soc. Jpn. **79**, 112001 (2010).

[2] C. Gros, Ann. Phys. **189**, 53–88 (1989).

[3] D. Tahara, M. Imada, Journal of the Physical Society of Japan **77**, 114701 (2008).

[4] T. Misawa, M. Imada, Phys. Rev. B **90**, 115137 (2014).

[5] S. Morita, R. Kaneko, M. Imada, J. Phys. Soc. Jpn. **84**, 024720 (2015).

[6] http://ma.cms-initiative.jp/ja/listapps/hphi.

[7] M. Wimmer: ACM Trans. Math. Software 38 30 (2012).

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