DSQSS Documentation

Release 2.0.0-DEV

DSQSS developers

Apr 18, 2019

CONTENTS

CHAPTER

ONE

ABOUT DSQSS

1.1 Overview

DSQSS is a program package for solving quantum many-body problems defined on lattices. It is based on the quantum Monte Carlo method in Feynman's path integral representation. It covers a broad range of problems written by flexible input files that define arbitrary unit cells in arbitrary dimensions, and arbitrary matrix elements of the interactions among arbitrary number of degrees of freedom.

For example, you can perform finite temperature calculation of XXZ spin model by specifying parameters such as dimension, size of lattice, anisotropic coupling constants, length of spin, strength of magnetic field, and temperature. You can calculate Bose-Hubbard model as well as quantum spin model. PMWA (Parallel Multi Worm Algorithm) suits for large-scale non-trivial parallel calculation by domain parallelization.

DSQSS consists of the following subpackages.

- serial (dla)
  1. Input files generator: `dla_pre`
  2. Hamiltonian file generator: `dla_hamgen`
  3. Lattice file generator: `dla_latgen`
  4. Algorithm file generator: `dla_alg`
  5. QMC engines (directed loop algorithm): `dla`
- non-trivial parallel (pmwa)
  1. Input files generator: `pmwa_pre`
  2. Lattice file generator: `lattgene_P`
  3. QMC engines (multiworm): `pmwa_H` (XXZ spin model), `pmwa_B` (Hard core Boson)

1.2 Developers

2018/10/19

- Yasuyuki Kato (Univ. of Tokyo)
- Naoki Kawashima (ISSP)
- Kota Sakakura (NEC)
- Takafumi Suzuki (Hyogo Univ.)
- Kenji Harada (Kyoto Univ.)

- Akiko Masaki (Hitachi Research Laboratory)
- Yuichi Motoyama (ISSP)
- Kazuyoshi Yoshimi (ISSP)

1.3 Collaborators

2018/10/19

- Tsuyoshi Okubo (Univ. of Tokyo)
- Takeo Kato (ISSP)

1.4 License

- GNU General Public License (GPL)

The users are kindly requested to acknowledge the usage of this software in their publication, if any, based on the software, and let the developers know its reference information.

Acknowledgment Sample

Numerical results in the present paper were obtained by the quantum Monte Carlo program DSQSS(https://github.com/qmc/dsqss/wiki). This package is distributed under GNU General Public License version 3 (GPL v3) or later.

1.5 Acknowledgment

Development of this software was and is being supported by K-computer project, post-K computer projects and "Project for advancement of software usability in materials science" by ISSP.

1.6 Contact

Write topics to GitHub's issues or send the e-mail to the following mailing list *dsqss-dev@issp.u-tokyo.ac.jp*

CHAPTER

TWO

HOW TO INSTALL

2.1 Requirements

- BLAS
- LAPACK
- (Optional) MPI (essential for PMWA)

2.2 Download

- Download zip file

  The latest version of DSQSS can be obtained from here .

- Use git

  Type the following command:

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

2.3 Directory structure

```
|-- CMakeLists.txt
|-- LICENSE
|-- README.md
|-- config
|   |-- gcc.cmake
|   `-- intel.cmake
|-- doc
|   |-- CMakeLists.txt
|   |-- en
|   `-- jp
|-- sample
|   |-- CMakeLists.txt
|   |-- dla
|   `-- pmwa
|-- src
|   |-- common
|   |-- dla
|   |-- pmwa
```

(continues on next page)

(continued from previous page)

```
|   `-- third-party
|-- test
|   |-- CMakeLists.txt
|   |-- dla
|   |-- pmwa
|   `-- tool
`-- tool
    |-- CMakeLists.txt
    |-- cmake
    |-- dsqss
    `-- setup.py
```

2.4 Install

The installation of DSQSS can be done by the following procedures. In the following, we assume the current directory is `dsqss.src`.

```
$ mkdir dsqss.build && cd dsqss.build
$ cmake ../ -DCMAKE_INSTALL_PREFIX=/path/to/install/to
$ make
```

Replace `/path/to/install/to` with the directory path where you want to install dsqss. It is noted that the default install directory is set as `/usr/local/bin` . When the build procedure failed, please try directly to set the compilers. For details, see https://github.com/issp-center-dev/HPhi/wiki/FAQ .

Each binary files for dsqss will be made in *src* folder. To check whether the binary files are correctly made or not, please type the following command:

```
$ make test
```

After seeing that all tests are passed, type the following command to install binary files:

```
$ make install
```

If the path for installation was changed, it is convenient to export the path:

```
$ export PATH="/path/to/install/to:$PATH"
```

CHAPTER

THREE

USAGE

DSQSS provides an input file creation tool `dsqss_pre.py` . This section briefly explains how to run DSQSS using this tool. In `DSQSS/DLA`, it is also possible for users to define complex models and execute them. For details, see *User's manual of DSQSS/DLA* .

3.1 Make an input file

To run `dsqss_pre.py`, a text-based input file is needed. An example of the input file is shown below.

```
[System]
solver = DLA
[Hamiltonian]
model_type = spin
M = 1
J = 1.0
F = 0.0
[Lattice]
lattice_type = square
D = 1
L = 8
Beta = 10
[Parameter]
NPRE = 1000
NTHERM = 1000
NMCS = 1000
NSET = 10
SEED = 31415
NVERMAX = 10000000
algfile = algorithm.xml
latfile = lattice.xml
outfile = sample.log
```

In the input file, specify the parameters that are divided into four categories. Each parameter is specified in the form `keyword = parameter`.

1. System section

   Set solvers with `solver` as the keyword. Solvers can be selected from `DLA` or `PMWA`.

2. Hamiltonian section

   Specify the system type (spin or boson) and the parameters to construct the Hamiltonian.

Parameter	Solver	Type	Default	Remarks
model_type	Common	str	spin	spin or boson.

- model_type = spin

Parameter	Solver	Type	Default	Remarks
M	DLA	int	1	The number of sub-spins on each site.
F	DLA	double	0.0	The magnetic field in the pair Hamiltonian. (= H/z if the field H is shared equally by all pairs, where $z = 4$ for a two-dimensional square lattice.).
J	DLA	double	1.0	The coupling constant J (positive for ferromagnets, negative for anti-magnets).
Jxy	PMWA	double	0.0	The coupling constant J_{xy} (positive for ferromagnets, negative for anti-magnets).
Jz	PMWA	double	0.0	The coupling constant J_z (positive for ferromagnets, negative for anti-magnets).
h	PMWA	double	0.0	The longitudinal magnetic field (not depend on z).
Gamma	PMWA	double	0.0	The transverse magnetic field (not depend on z).

- model_type = boson

Parameter	Solver	Type	Default	Remarks
t	Common	double	1.0	The hopping constant.
U	Common	double	0.0	The on-site interaction (positive for repulsive).
V	Common	double	0.0	The nearest neighbor interaction (positive for repulsive).
M	DLA	int	1	The maximum number of bosons on each site.
F	DLA	double	0.0	The chemical potential in the pair Hamiltonian (= H/z if the field H is the field per site and H is shared equally by all pairs, e.g., $F = H/4$ for a square lattice.)
mu	PMWA	double	0.0	The chemical potential.
Gamma	PMWA	double	0.0	The source term (the coefficient of $b_i^\dagger + b_i$).

For details of the parameters constituting the Hamiltonian, see the section for input files of hamgen_B or hamgen_H for DSQSS/DLA. For DSQSS/PMWA , see the section for input file.

3. Lattice section

   This section specifies the lattice type and inverse temperature.

Parameter	Type	Default	Remarks
lattice_type	str	square	Select square or triangular for DLA. For PMWA, only square can be selected.
D	int		The number of dimension.
L	int		The liner size of the lattice. Specify the size of the D dimensional space, separated by , for example, in the case of a lattice of 2D 2×4, specify it as L = 2, 4.
Beta	double	10.0	Inverse temperature
NLdiv	int	1	(for DSQSS/PMWA): The devision number for the lattice.
NBdiv	int	1	(for DSQSS/PMWA): The devision number for Beta .

4. Parameter section

   In this section, the calculation condition is specified. Set the parameters using keywords common to the input files of DSQSS / DLA and DSQSS / PMWA. Please refer to the input file of each solver for details of defined parameters.

3.2 Execution

After making the input file, input files for the solver are generated by typing the following command (input file name is set as "std.in" in the following).

```
$ dsqss_pre.py -i std.in
```

For DSQSS/DLA , algorithm.xml, hamiltonian.xml, lattice.xml, param.in are generated. For DSQSS/PMWA, lattice.xml, param.in are generated. To execute the solver, see the end line of standard output of dsqss_pre.py

:

Please type: xxxxxx

where xxxxxx is the command for the execution (for example, DLA_H param.in). To use MPI, add mpirun -np 8 in the front of xxxxxx (8 is the number of the processes to be used for parallelization).

DSQSS/DLA specifies the number of parallel random numbers, DSQSS/PMWA specifies the product of the number of parallel random numbers and the total number of divisions (product of space division number and imaginary time division number) as the number of processes. Details of the output files after calculations are described in the tutorial / output result of each solver, so please refer to that.

3.3 Flow of Monte Carlo calculation

In Fig. 3.1 , the flow of Monte Carlo calculation is shown.

DSQSS defines 1 MC step from generation of worm head pair to disappearance and defines 1 MC sweep as N_{cyc} MC step (even if generation of worm head pair fails, it is counted as 1 MC step.) The value of N_{cyc} is determined by the initial NPRE MC step.

After determining N_{cyc} , Simulation of the NTHERM MC sweep is performed as an initial relaxation phase, and the simulation of NMCS MC sweep continues as a physical quantity measurement phase.

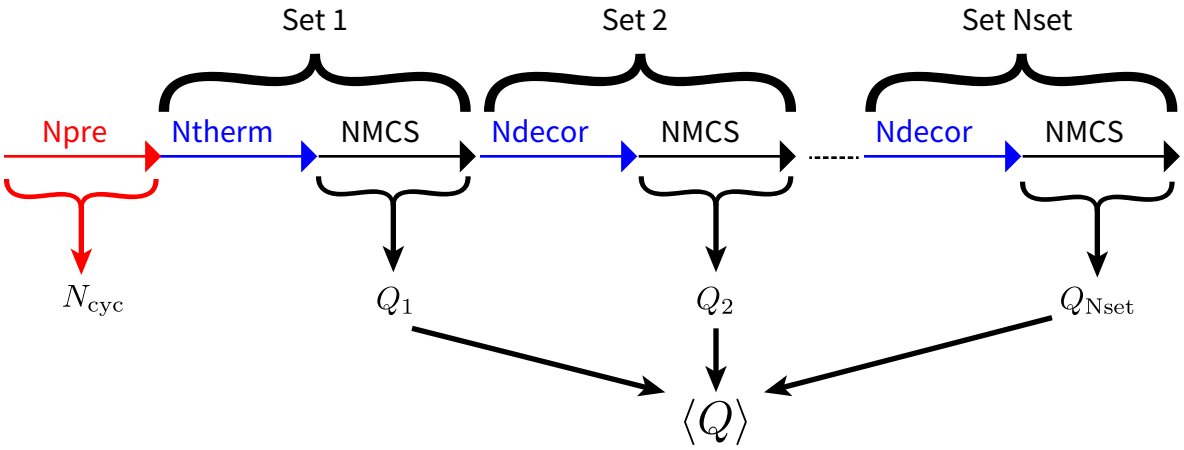

Fig. 3.1: Schematic figure for the flow of Monte Carlo calculation and the parameters for Monte Carlo steps

A `NDECOR` MC sweep simulation is performed as an autocorrelation reduction phase between one physical quantity measurement phase and the next physical quantity measurement process phase.

One set consists of two phases, the initial relaxation phase and the physical quantity measurement phase, or the autocorrelation reduction phase and the physical quantity measurement phase, and the whole simulation includes `NSET` sets.

Expected value of physical quantity $\langle Q \rangle$ and error σ_Q can be obtained as the mean and standard error of the physical quantity obtained from each of the `NSET` sets.

CHAPTER

FOUR

TUTORIAL OF DSQSS/DLA

4.1 What's DSQSS/DLA?

DSQSS/DLA is a solver package based on the world line Monte Carlo method with the directed loop algorithm. This can simulate many quantities, e.g. magnetization and susceptibility, of any model on any lattice if free from negative sign problem. This comes with some utility tools which generate input files for some models and lattices; Heisenberg model, Bose-Hubbard model, hypercubic lattice, and triangular lattice.

In this chapter, DSQSS is installed to the directory where the environment variable `$DSQSS_ROOT` points.

4.2 Energy calculation of the antiferromagnetic Heisenberg dimer by DSQSS/DLA

This tutorial gives how to use DSQSS/DLA through a calculation the energy of the $S = 1/2$ antiferromagnetic Heisenberg dimer $\mathcal{H} = -J\vec{S}_1 \cdot \vec{S}_2$.

DSQSS/DLA calculation has the following three parts:

1. Prepare the input files
2. Perform QMC calculation
3. Analyze the result

4.2.1 Prepare the input files

DSQSS/DLA requires the following input files:

1. Parameter file
2. lattice file
3. algorithm file

`dsqss_pre.py` is a utility tool to generate these files from one textfile such as the following (sample/dla/01_spindimer/std.in)

```
[System]
solver = DLA
[Hamiltonian]
model_type = spin
M =  1                  # S=1/2
```

(continues on next page)

(continued from previous page)

```
J = -0.5                # coupling constant, negative for AF, not 1 but 1/2 due to PBC
F = 0.0                 # magnetic field
[Lattice]
lattice_type = square   # hypercubic, periodic
D = 1                   # dimension
L = 2                   # number of sites along each direction
Beta = 100              # inverse temperature
[Parameter]
nset = 5                # set of Monte Carlo sweeps
npre = 10               # MCSteps to estimate hyperparameter
ntherm = 10             # MCSweeps for thermalization
nmcs = 10               # MCSweeps for measurement
```

Note that the coupling constant J is set not to -1.0 but -0.5 since this tool generates a lattice under the periodic boundary condition and in this case the lattice has two sites and two bonds in results.

Give this file to dsqss_pre.py as

```
$ python $DSQSS_ROOT/bin/dsqss_pre.py -i std.in
```

This generates the following four files: a parameter file `param.in`, a lattice file `lattice.xml`, an algorithm file `algorithm.xml`, and an auxiliary file for the algorithm file `hamiltonian.xml`.

4.2.2 Perform QMC calculation

Once input files are prepared, you can perform a quantum Monte Carlo calculation based on the directed loop algorithm (dla) by DSQSS/DLA.

```
$ $DSQSS_ROOT/bin/dla_H param.in
```

`dla_H` is a dla solver for spin systems. If you want to deal with Bose-Hubbard models, please use `dla_B`.

You can perform random number parallelization by using MPI.[1]

```
$ mpiexec -np 4 $DSQSS_ROOT/bin/dla_H param.in
```

By the above command, the total number of Monte Carlo samples times by four (equals to the number of process) and the obtained statistical error is expected to reduce to half (equals to the inverse square root of the number of process).

4.2.3 Analyze the result

The result of the calculation is written into a text file `sample.log`. Since the energy per site is recorded with a name `ene`, you can for example draw this by the `grep` command by the following.

```
$ grep ene sample.log
R ene = -3.75780000e-01 8.89848302e-04
```

The two figures stand for the expectation value and the statistical error, respectively. The result value is compatible with the exact solution, $-3|J|/8 = -0.375|J|$, within the statistical error.

[1] After finishing DSQSS/DLA, the OpenMPI on macOS may say an error message, `No such file or directory (errno 2)`. You can ignore this error safely. If you're annoyed by it, please put an extra option `--mca shmem posix` to `mpiexec`.

4.3 Magnetic Susceptibility of antiferromagnetic spin chains

In this tutorial, we will calculate the temperature dependence of the magnetic susceptibility for two kinds of antiferromagnetic spin chains with the local spin length $S = 1/2, 1$.

The following Python script (sample/dla/02_spinchain/exec.py) performs DSQSS/DLA work-flow for each parameter (combination of spin length and temperature) automatically.

```
import sys
import os.path
import subprocess

bindir = sys.argv[1] if len(sys.argv) > 1 else ''

name = 'xmzu'
Ms = [1,2]
Ts = [0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.25, 1.5, 1.75, 2.0]

for M in Ms:
    output = open('{}_{}.dat'.format(name,M), 'w')
    for i,T in enumerate(Ts):
        with open('std_{}_{}.in'.format(M,i), 'w') as f:
            f.write('''
    solver = DLA
    model_type = spin
    J = -1
    F = 0.0
    lattice_type = square
    D = 1
    L = 30
    nset = 5
    ntherm = 1000
    ndecor = 1000
    nmcs = 1000
    ''')
            f.write('M = {}\n'.format(M))
            f.write('beta = {}\n'.format(1.0/T))
            f.write('outfile = res_{}_{}.dat\n'.format(M,i))
        cmd = [os.path.join(bindir, 'dsqss_pre.py'),
               '-p', 'param_{}_{}.in'.format(M,i),
               '-i', 'std_{}_{}.in'.format(M,i)]
        subprocess.call(cmd)
        cmd = [os.path.join(bindir, 'dla_H'), 'param_{}_{}.in'.format(M,i)]
        subprocess.call(cmd)
        with open('res_{}_{}.dat'.format(M,i)) as f:
            for line in f:
                if not line.startswith('R'):
                    continue
                words = line.split()
                if words[1] == name:
                    output.write('{} {} {}\n'.format(T, words[3], words[4]))
```

This script receives the binary directory as an argument (if an environment variable `$DSQSS_ROOT` is set correctly, the argument can be omitted).

```
$ python exec.py $DSQSS_ROOT/bin
```

The result of $S = 1/2, 1$ will be written to `xmzu_1.dat` and `xmzu_2.dat`, respectively (Fig. 4.1). The $S = 1/2$

chain is gapless and so the magnetic susceptibility remains finite at absolute zero (note that in the simulation the finite size effect opens energy gap). On the other hand the magnetic susceptibility of the $S = 1$ chain drops to zero at finite temperature due to the spin gap.

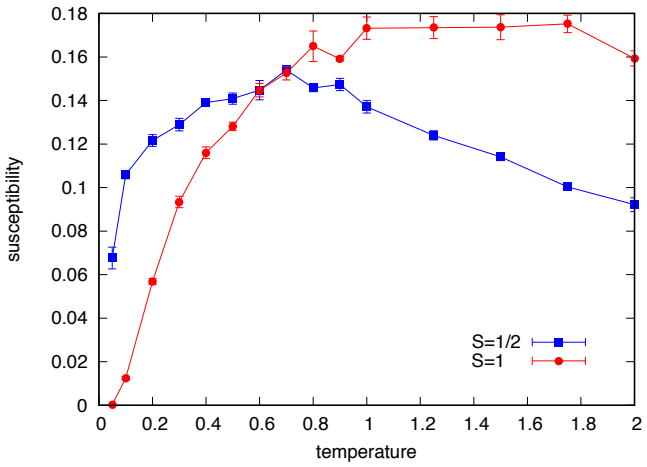

Fig. 4.1: Temperature v.s. susceptibility curves for $S = 1/2$ (blue) and $S = 1$ (red) antiferromagnetic Heisenberg chain.

4.4 Number density of the hardcore Bosons on a square lattice

In this tutorial, we will calculate the chemical potential dependence of the number density of the hardcore Bose-Hubbard model with the nearest neighbor repulsive on a 8×8 square lattice.

The following Python script (sample/dla/03_bosesquare/exec.py) performs DSQSS/DLA work-flow for each parameter (chemical potential) automatically.

```
import sys
import os.path
import subprocess

bindir = sys.argv[1] if len(sys.argv) > 1 else ''

name = 'amzu'
mus = [-4.0, -2.0, 0.0, 2.0, 2.5, 3.0, 6.0, 9.0, 9.5, 10.0, 12.0, 14.0]

output = open('{}.dat'.format(name), 'w')

for i,mu in enumerate(mus):
    with open('std_{}.in'.format(i), 'w') as f:
        f.write('''
solver = DLA
model_type = boson
M = 1
J = 1
U = 0
V = 3
beta = 10.0
lattice_type = square
```

(continues on next page)

(continued from previous page)

```
D = 2
L = 8,8
nset = 4
ntherm = 100
ndecor = 100
nmcs = 100
''')
        f.write('F = {}\n'.format(mu/4))
        f.write('algfile = algorithm_{}.xml\n'.format(i))
        f.write('outfile = res_{}.dat\n'.format(i))
    cmd = [os.path.join(bindir, 'dsqss_pre.py'),
           '-p', 'param_{}.in'.format(i),
           '-i', 'std_{}.in'.format(i)]
    subprocess.call(cmd)
    cmd = [os.path.join(bindir, 'dla_B'), 'param_{}.in'.format(i)]
    subprocess.call(cmd)
    with open('res_{}.dat'.format(i)) as f:
        for line in f:
            if not line.startswith('R'):
                continue
            words = line.split()
            if words[1] == name:
                output.write('{} {} {}\n'.format(mu, words[3], words[4]))
```

This script receives the binary directory as an argument (if an environment variable `$DSQSS_ROOT` is set correctly, the argument can be omitted).

```
$ python exec.py $DSQSS_ROOT/bin
```

The result is written to `amzu.dat` (Fig. 4.2). You can see a density plateau around $\mu = 6$. In this region, a checker board solid phase due to repulsive interaction appears.

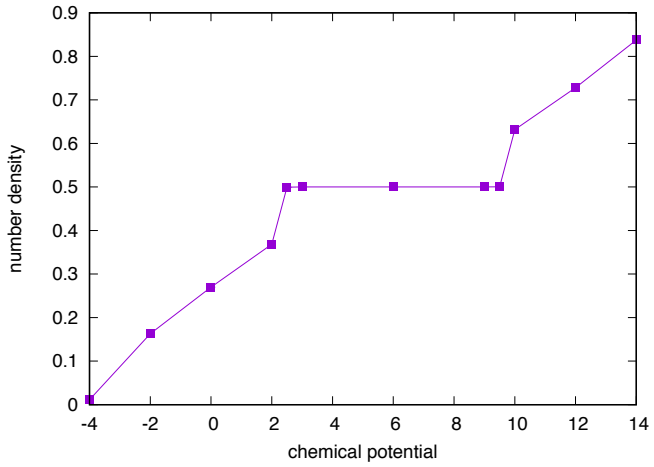

Fig. 4.2: Chemical potential dependence of number density of repulsive hardcore bosons.

CHAPTER

FIVE

USER'S MANUAL OF DSQSS/DLA

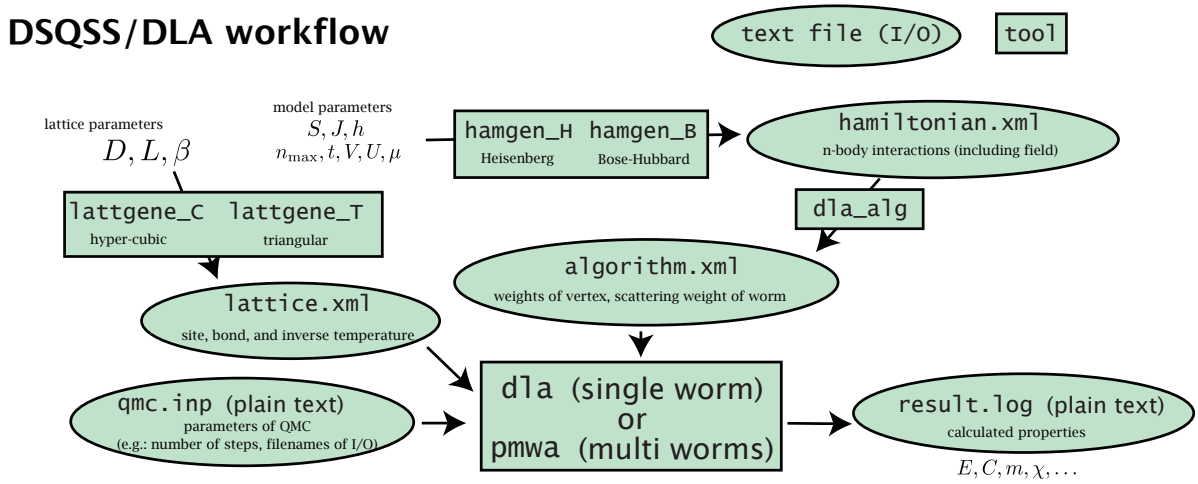

Fig. 5.1: Work-flow of DSQSS/DLA

5.1 Input files for DSQSS/DLA

5.1.1 The list of input files

qmc.inp	Parameter list for the simulation, e.g., the number of Monte Carlo sets.
lattice.xml	Definition of the lattice.
algorithm.xml	Definition of the algorithm (e.g., scattering rate of a worm).
sf.xml	Indication of wave vectors for structure factors. (optional)
cf.xml	Indexing directions between all the sites. (optional)

5.1.2 Parameter file `qmc.inp`

The parameter file is a plain-text file with the following format,

- One line stands for one parameter by the key-value style, `<name> = <value>`.
- Cases are insensitive except for file names.

- White spaces and blank lines are ignored.
- Parts between "#" symbol and the linebreak are ignored as comments.

The list of parameters are the following,

name	type	default	description
beta	double	–	Inverse temperature.
npre	int	1000	The number of Monte Carlo steps in the pre-calculation phase where the number of creation trials of a pair of worms in one Monte Carlo sweep is defined.
ntherm	int	1000	The number of Monte Carlo sweeps to thermalize the system.
ndecor	int	1000	The number of Monte Carlo sweeps to reduce autocorrelation time between two preceding sets.
nmcs	int	1000	The number of Monte Carlo sweeps to calculate mean values of observables.
nset	int	10	The number of Monte Carlo sets.
simulationtime	double	0.0	Simulation time in second.
seed	int	198212240	The seed of the random number generator.
nvermax	int	10000	The maximum number of vertices.
nsegmax	int	10000	The maximum number of world-line segments.
algfile	string	algorithm.xml	The filename of an algorithm file.
latfile	string	lattice.xml	The filename of a lattice file.
sfinpfile	string	–	A structure factor file. If it is an empty string, structure factors will not be calculated.
cfinpfile	string	–	A real space temperature Green's function file. If it is an empty string, real space temperature Green's functions will not be calculated.
ckinpfile	string	–	A momentum space temperature Green's function file. If it is an empty string, momentum space temperature Green's functions will not be calculated.
outfile	string	sample.log	The name of the main result file.
sfoutfile	string	sf.dat	The name of the structure factor result file.
cfoutfile	string	cf.dat	The name of the real space temperature Green's function output file.
ckoutfile	string	ck.dat	The name of the momentum space temperature Green's function output file.
runtype	int	0	Method. This remains for backward compatibility.

- About simulationtime
  - When simulationtime > 0.0
    - **DSQSS/DLA loads the checkpoint file and resumes the simulation if the checkpoint file exists.**
      - If not, DSQSS/DLA starts a new simulation.
    - After the time specified by "simulationtime" (in seconds) has elapsed, DSQSS/DLA saves the state of the simulation into the checkpoint file and halts the simulation.
    - The name of the checkpoint file is that of the main result file with a suffix ".cjob".
  - When simulationtime <= 0.0
    - The checkpoint file is ignored. DSQSS/DLA never saves nor loads it.

5.1.3 Lattice file `lattice.xml`

A lattice file is a textfile written in XML format. This defines timespace to be simulation, for example, the number of sites, the connections between sites, the inverse temperature, and so on. This file can be very complicated, so DSQSS has a utility tool `lattgene` to generate lattice files describing common lattices, such as a hypercubic lattice.

The lattice file has a unique element named "Lattice". The other elements belong to "Lattice" as children.

Lattice The root element.

Lattice/Comment (Optional) Comment. DSQSS ignores this element.

Lattice/Dimension The dimension of the lattice.

Lattice/LinearSize The size of the lattice in units of the unitcell. This takes space-separated positive integers as many as specified by "Lattice/Dimension".

```
<LinearSize> 3 4 </LinearSize>
# unitcells are arranged in 3x4.
```

Lattice/NumberOfCells The number of unitcells.

Lattice/NumberOfSites The number of sites.

Lattice/NumberOfInteractions The number of the interactions. When all the interactions are two-body ones, this is nothing but the number of bonds.

Lattice/NumberOfSiteTypes The number of site types.

Lattice/NumberOfInteractionTypes The number of interaction types.

Lattice/BondDimension Parameter for the winding number.

Lattice/NumberOfEdgeInteractions Parameter for the Winding number. The number of bonds connecting sites over the lattice's boundary.

Lattice/S Site information. "Lattice" should includes this element as many as the number specified by "Lattice/NumberOfSites". This takes three positive integers, "index of site", "site type", and "measure type". The detail of site type is defined in an algorithm file.

```
<S> 3 0 1 </S>
# the site with index 3 has the site type of 0 and the measure type of 1.
```

Lattice/I Interaction information. "Lattice" should includes this element as many as the number specified by "Lattice/NumberOfInteractions". This takes space-separated integers, "index of the interaction", "interaction type", "the number of sites involved in the interaction", "indices of involved sites". The details of interaction type, e.g., the strength, are defined in an algorithm file. The order of the indices of sites should be compatible with the order of sites specified in "Algorithm/Vertex/InitialConfiguration" in the algorithm file.

```
<I> 5 1 2 8 12 </I>
# the interaction with index 5 has the interaction type of 1 and connects 2 sites,
↪ 8 and 12.
```

5.1.4 Algorithm file `algorithm.xml`

An algorithm file is a textfile written in XML format. This defines the details of interactions, for example, the scattering probability of a worm head. This file can be very complicated, so DSQSS has a utility tool `dla_alg` to generate algorithm files from more simple file, the Hamiltonian file introduced later.

The algorithm file has a unique element named "Algorithm". The other elements belong to "Algorithm" as children.

Algorithm The root element. This has children, "General", "Site", "Interaction", and "Vertex".

Algorithm/Comment (Optional) Comment. DSQSS ignores this.

Algorithm/General General parameters such as the number of site types. This has children, "NSType", "NIType", "NVType", "NXMax", and "WDiag".

```
<Algorithm>
  <General>
    <NSType>  1 </NSType>
    <NIType>  1 </NIType>
    <NVType>  2 </NVType>
    <NXMax>   2 </NXMax>
    <WDiag>   0.25 </WDiag>
  </General>
  ...
</Algorithm>
```

Algorithm/General/NSType The number of site types.

Algorithm/General/NIType The number of interaction types.

Algorithm/General/NVType The number of vertex types.

Algorithm/General/NXMax The maximum number of states on a site. For example, $2S + 1$ for a spin system with local spin S.

Algorithm/General/WDiag User can use this value for user's own purpose in "measure_specific.cc". In the original "measure_specific.cc" uses this value as a coefficient to measure correlation functions from the length of worms.

Algorithm/Site This defines a site type, for example, the weight of worm heads on a site. This has children "SType", "NumberOfStates", "VertexTypeOfSource", and "InitialConfiguration".

```
<Algorithm>
  ...
  <Site>
    <STYPE> 0 </STYPE>
    <NumberOfStates> 2 </NumberOfStates>
    <VertexTypeOfSource> 0 </VertexTypeOfSource>
    <InitialConfiguration>
       ...
    </InitialConfiguration>
    <InitialConfiguration>
       ...
    </InitialConfiguration>
  </Site>
  ...
</Algorithm>
```

Algorithm/Site/SType The index of site type.

Algorithm/Site/NumberOfStates The number of states of the site.

Algorithm/Site/VertexTypeOfSource The index of the vertex to be inserted here.

Algorithm/Site/InitialConfiguration The process of pair creation/annihilation of worm heads. This has children, "State", "NumberOfChannels", and "Channel"

```
<Algorithm>
  ...
  <Site>
```

(continues on next page)

(continued from previous page)

```
    ...
    <InitialConfiguration>
      <State> 0 </State>
      <NumberOfChannels> 2 </NumberOfChannels>
      <Channel> 0 1 0.5 </Channel>
      <Channel> 1 1 0.5 </Channel>
    </InitialConfiguration>
    ...
  </Site>
  ...
</Algorithm>
```

Algorithm/Site/InitialConfiguration/State The state index of the site without worms (before creation or after annihilation).

Algorithm/Site/InitialConfiguration/NumberOfChannels The number of the channels (result of creation/annihilation).

Algorithm/Site/InitialConfiguration/Channel Channels. This takes two integers and one floating number.

- First figure denotes the direction of the worm head (0 for negative and 1 for positive in the imaginary time direction).
- Second figure denotes the state between worms.
- Third figure denotes the probability of this channel.

If the result has no worm heads, let both the first and the second integers be -1.

Algorithm/Interaction This defines an interaction. This has children, "IType", "VType", "NBody", "EBase", "VertexDensity", and "Sign".

```
<Algorithm>
  ...
  <Interaction>
    <IType> 0 </IType>
    <VType> 1 </VType>
    <NBody> 2 </NBody>
    <EBase> 0.125 </EBase>
    <VertexDensity> 0 0 0.25 </VertexDensity>
    <VertexDensity> 1 1 0.25 </VertexDensity>
    <Sign> 0 1 1 0 -1.0 </Sign>
    <Sign> 1 0 0 1 -1.0 </Sign>
  </Interaction>
  ...
</Algorithm>
```

Algorithm/Interaction/IType The index of the interaction.

Algorithm/Interaction/VType The index of the vertex to be inserted.

Algorithm/Interaction/NBody The number of sites involved in this interaction. An onebody interaction such as the Zeeman term has 1 and a twobody interaction such as the exchange coupling has 2. Three or higher body interaction can be treated.

Algorithm/Interaction/EBase The offset of the local energy. This value does not contribute to the simulation, but to the value of energy in the final result.

Algorithm/Interaction/VertexDensity The density of vertex to be inserted. This takes integers as many as "Algorithm/Interaction/NBody" and one preceding floating number. The integers denote the states of sites (the order

should be compatible with the order of sites in "I" of the lattice file). The last floating number represents the density.

Algorithm/Interaction/Sign The sign of the local weight, $\mathrm{Sgn}(\langle f| -\mathcal{H}|i\rangle)$. This takes integers as many as $2\times$ "Algorithm/Interaction/NBody" and one preceding floating number. The integers denote the states of sites before and after applying the local Hamiltonian. The last floating number represents the sign. If the sign is equal to 1.0:, this element (`<Sign> ... </Sign>`) can be omitted.

For example, `<Sign> 0 1 1 0 -1.0 </Sign>` means $\langle 10|\,(-\mathcal{H})\,|01\rangle < 0$.

Algorithm/Vertex This defines a vertex. This has children, "VType", "VCategory", "NBody", "NumberOfInitialConfigurations", and "InitialConfiguration". Vertices belongs to a category specified by "Algorithm/Vertex/VCategory".

```
<Algorithm>
  ...
  <Vertex>
    <VTYPE> 0 </VTYPE>
    <VCATEGORY> 1 </VCATEGORY>
    <NBODY> 1 </NBODY>
    <NumberOfInitialConfigurations> 4 </NumberOfInitialConfigurations>
    <InitialConfiguration>
      ...
    </InitialConfiguration>
    ...
    <InitialConfiguration>
      ...
    </InitialConfiguration>
  </Vertex>
  ...
</Algorithm>
```

Algorithm/Vertex/VType The index of the vertex.

Algorithm/Vertex/VCategory

0. Boundary of imaginary time. Users need not define this.
1. Worm tail.
2. Interaction.

Algorithm/Vertex/NBody The number of sites involved.

Algorithm/Vertex/NumberOfInitialConfigurations The number of initial states.

Algorithm/Vertex/InitialConfiguration

This defines scattering results of a worm head for each initial states. "Algorithm/Vertex" should has this elements as many as the number specified by "Algorithm/Vertex/NumberOfInitialConfigurations". This has children, "State", "IncomingDirection", "NewState", "NumberOfChannels", "Channel".

```
<Algorithm>
  ...
  <Vertex>
    ...
    <InitialConfiguration>
      <State>  1 0 0 1 </State>
      <IncomingDirection> 0 </IncomingDirection>
      <NewState> 0 </NewState>
```

(continues on next page)

(continued from previous page)

```
      <NumberOfChannels> 1 </NumberOfChannels>
      <Channel>    3    0       1.0000000000000000 </Channel>
    </InitialConfiguration>
    ...
  </Vertex>
  ...
</Algorithm>
```

This example represents the following scenario;

- Initial states of bottom-left(0), top-left(0), bottom-right(2), and top-right(3) are 1, 0, 0, and 1, respectively.
- A worm head comes from bottom-left(0) and changes the state of this leg to 0.
- The worm head will be scattered to leg(3) and the state of outgoing leg will be changed to 0 with the probability 1.

Algorithm/Vertex/InitialConfiguration/State The initial states of the legs of the vertex. Since the number of the legs is as twice as the number specified by "Algorithm/Vertex/NBody", say m, this takes $2m$ integers. Legs are in the same order as the corresponding sites. For two legs on the same site, the leg with the smaller imaginary time comes first.

Algorithm/Vertex/InitialConfiguration/IncomingDirection The index of the leg from which a worm head comes.

Algorithm/Vertex/InitialConfiguration/NewState The state of the "Algorithm/Vertex/InitialConfiguration/IncomingDirection" leg after a worm head comes.

Algorithm/Vertex/InitialConfiguration/NumberOfChannels The number of scattering channels (final results).

Algorithm/Vertex/InitialConfiguration/Channel A scattering channel. This takes two integers and one floating number.

- First figure denotes the **index** of the leg where the scattered worm head goes out.
- Second figure denotes the **state** of the leg where the scattered worm head goes out after the scattering.
- Last figure denotes the probability of this channel.

For the special case, the pair-annihilation of worm heads, let both the first and the second integer be -1.

5.1.5 Hamiltonian file `hamiltonian.xml`

A Hamiltonian file is a textfile written in XML format. This defines the local Hamiltonians, e.g., a bond Hamiltonian. This file is used as an input of `dla_alg` in order to generate `algorithm.xml` . DSQSS has utility tools `hamgen_H` and `hamgen_B` to generate hamiltonian files describing the Heisenberg spin model and the Bose-Hubbard model.

The Hamiltonian file has a unique element named "Hamiltonian". The other elements belong to "Hamiltonian" as children.

Hamiltonian The root element. This has children, "General", "Site", "Source", and "Interaction".

Hamiltonian/General General parameters such as the number of site types. This has children, "NSTYPE", "NITYPE", "NXMAX", and "Comment".

```
<Hamiltonian>
   <General>
     <Comment> SU(2) Heisenberg model with S=1/2 </Comment>
     <NSTYPE> 1 </NSTYPE>
     <NITYPE> 1 </NITYPE>
     <NXMAX>  2 </NXMAX>
```

(continues on next page)

(continued from previous page)

```
   </General>
  ...
</Hamiltonian>
```

Hamiltonian/General/Comment (Optional) Comment. DSQSS ignores this.

Hamiltonian/General/NSTYPE The number of site types.

Hamiltonian/General/NITYPE The number of interaction types.

Hamiltonian/General/NXMAX The maximum number of states on a site. For example, $2S + 1$ for a spin system with local spin S.

Hamiltonian/Site This defines a site type, for example, the number of states. This has children "STYPE", "TTYPE", and "NX".

```
<Hamiltonian>
  ...
  <Site>
    <STYPE> 0 </STYPE>
    <TTYPE> 0 </TTYPE>
    <NX>   2 </NX>
  </Site>
  ...
</Hamiltonian>
```

Hamiltonian/Site/STYPE The index of site type.

Hamiltonian/Site/TTYPE The index of the source type (type of pair creation/annihilation of worm-heads.)

Hamiltonian/Site/NX The number of states of the site.

Hamiltonian/Source This defines a source type, that is, the pair-creation/annihilation of worm-heads. This has children "TTYPE", "STYPE", and "Weight".

```
<Source>
  <TTYPE> 0 </TTYPE>
  <STYPE> 0 </STYPE>
  <Weight> 0 1       0.5000000000000000 </Weight>
  <Weight> 1 0       0.5000000000000000 </Weight>
</Source>
```

Hamiltonian/Source/TTYPE The index of the source type.

Hamiltonian/Source/STYPE The index of the site type.

Hamiltonian/Source/Weight The weight of the creation/annihilation operator. This takes two integers and one floating number. The integers denote the states of the site before and after applying the operator, respectively. The floating number denotes the matrix element.

For example, `0 1 0.5` means $\langle 1|\mathcal{H}|0\rangle = 0.5$.

Hamiltonian/Interaction This defines an interaction type. This has children "ITYPE", "STYPE", "NBODY", and "Weight".

```
<Hamiltonian>
  ...
  <Interaction>
    <ITYPE> 0 </ITYPE>
    <NBODY> 2 </NBODY>
```

(continues on next page)

(continued from previous page)

```
    <STYPE> 0 0 </STYPE>
    <Weight> 0 0 0 0      -0.2500000000000000 </Weight>
    <Weight> 1 1 0 0       0.2500000000000000 </Weight>
    <Weight> 1 0 0 1      -0.5000000000000000 </Weight>
    <Weight> 0 1 1 0      -0.5000000000000000 </Weight>
    <Weight> 0 0 1 1       0.2500000000000000 </Weight>
    <Weight> 1 1 1 1      -0.2500000000000000 </Weight>
  </Interaction>
  ...
</Hamiltonian>
```

Hamiltonian/Interaction/ITYPE The index of the interaction type.

Hamiltonian/Interaction/NBODY The number of sites involved in this interaction. An onebody interaction such as the Zeeman term has 1 and a twobody interaction such as the exchange coupling has 2. Three or higher body interaction can be treated.

Hamiltonian/Interaction/ITYPE The indices of sites involved in this interaction. This takes NBODY integers.

Hamiltonian/Interaction/Weight The matrix elements of the local Hamiltonian.

This takes integers as many as $2\times$ NBODY and one preceding floating number. The integers denote the states of sites before and after applying the local Hamiltonian. The last floating number denotes the matrix element multiplied by -1.

For example, `0 0 1 1 0.25` means $\langle 01|-\mathcal{H}|01\rangle = 0.25$ and `0 1 1 0 -0.5` means $\langle 10|-\mathcal{H}|01\rangle = -0.5$.

5.1.6 Structure factor file `sf.xml`

A structure factor file is a textfile written in a XML-like format. This defines wave vectors and the discretization of imaginary time to calculate the dynamical structure factor

$$S^{zz}(\vec{k},\tau) \equiv \left\langle M^z(\vec{k},\tau)M^z(-\vec{k},0)\right\rangle - \left\langle M^z(\vec{k},\tau)\right\rangle \left\langle M^z(-\vec{k},0)\right\rangle .$$

DSQSS has a utility tool to generate a structure factor file, `sfgene`.

A structure factor file has only one element, "StructureFactor", and the other elements are children of this.

StructureFactor The root element. This has children, "Ntau", "NumberOfElements", "CutoffOfNtau", "NumberOfInverseLattice", and "SF".

StructureFactor/Comment (Optional) Comment. DSQSS ignores this.

StructureFactor/Ntau The number of discretization of the imaginary time axis.

StructureFactor/CutoffOfNtau The maximum of the imaginary time distance of the dynamical structure factor, τ. This takes an integer in $0, 1, \ldots, \mathrm{Ntau}$.

StructureFactor/NumberOfInverseLattice The number of wave vectors, $\vec{k}$

StructureFactor/NumberOfElements The number of the combination of wave vectors and sites.

StructureFactor/SF The phase factor $z = \exp \vec{r}\cdot\vec{k}$ for a pair of a wave vector and a site. This takes four figures, "$\mathrm{Re}z$", "$\mathrm{Im}z$", "the index of the site", "the index of the wave vector". "StructureFactor" should has this elements as many as the number specified by "StructureFactor/NumberOfElements".

5.1.7 Real space temperature Green's function file `cf.xml`

Real space temperature Green's function file is a textfile written in a XML-like format. This defines relative coordinate between two sites, $\vec{r}_{ij}$, to calculate real space temperature Green's function,

$$G(\vec{r},\tau) \equiv \frac{1}{N^2}\sum_{i,j}\left\langle M_i^+(\tau)M_j^-\right\rangle \delta(\vec{r}-\vec{r}_{ij}).$$

More precisely, this groups all the pair of sites by the relative coordinates.

DSQSS has a utility tool to generate a real space temperature Green's function file, `cfgene`.

A real space temperature Green's function file has only one element, "CorrelationFunction", and the other elements belong to this as children.

CorrelationFunction The root element. This has children, "Ntau", "NumberOfKinds", and "CF".

CorrelationFunction/Comment (Optional) Comment. DSQSS ignores this.

CorrelationFunction/Ntau The number of discretization of the imaginary time axis.

CorrelationFunction/NumberOfKinds The number of relative coordinates.

CorrelationFunction/CF This takes three integers, "the index of the relative coordinate", "the index of the site i", and "the index of the site j". "CorrelationFunction" should has this elements as many as the number specified by "CorrelationFunction/NumberOfKinds".

5.1.8 Momentum space temperature Green's function file `ck.xml`

A momentum space temperature Green's function file is a textfile written in a XML-like format. This defines wave vectors and the discretization of imaginary time to calculate the momentum space temperature Green's function,

$$G(\vec{k},\tau) \equiv \left\langle M^+(\vec{k},\tau)M^-(-\vec{k},0)\right\rangle.$$

Since this file has the format as same as that of the structure factor file including the names of elements, users can use the same file.

5.2 Utility tools to generate the input files of DSQSS/DLA

DSQSS/DLA needs several XML-formatted input files: lattice file, algorithm file, structure file, real space temperature Green's function file, and momentum space temperature Green's function file. By writing these files, you can simulate any model on any lattice. Since they are a little complicated, DSQSS/DLA prepares utility tools for generating them.

5.2.1 Hypercubic lattice generator `lattgene_C`

`lattgene_C` is a utility tool to generate a lattice file describing a D dimensional hypercubic lattice with the periodic boundary condition.:

```
$ lattgene_C [-o filename] D L1 L2 ... LD
```

The meaning of parameters are following:

`D` Dimension of lattice

`L1 L2 ... LD` Linear length of lattice in each dimension.

filename Name of lattice file (default: `lattice.xml`).

Example:

```
## Chain with 8 sites.
$ lattgene_C 1 8

## Square lattice with 6x6 sites.
## Name of the generated file is lat.xml
$ lattgene_C -o lat.xml 2 6 6
```

5.2.2 Triangular lattice generator `lattgene_T`

`lattgene_T` is a utility tool to generate a lattice file describing a triangular lattice with the periodic boundary condition.

```
$ lattgene_T [-o filename] L1 L2
```

The meaning of parameters are following:

L1 L2 Linear length of lattice in each dimension.

filename Name of lattice file (default: `lattice.xml`).

Example:

```
## Triangular lattice with 6x6 sites.
$ lattgene_T 1 6
```

5.2.3 Heisenberg spin Hamiltonian generator `hamgen_H`

`hamgen_H` is a utility tool generating a hamiltonian file describing Heisenberg spin model

```
$ hamgen_H [-o filename] M J F
```

The meaning of parameters is following:

M Twice as the length of the local spin, $2S$

J The exchange interaction. Positive for ferromagnetic and negative for antiferromagnetic.

F The magnetic field on a site per a bond connected to the site, $F = h/z$, where z is the coordination number, for example, $z = 4$ for the square lattice.

filename Name of hamiltonian file (default: `hamiltonian.xml`).

Example:

```
## S=1/2 antiferromagnetic Heisenberg model without magnetic field.
$ hamgen_H 1 -1.0 0.0

## S=1 ferromagnetic Heisenberg model with magnetic field.
## Name of the generated file ham.xml
$ hamgen_H -o ham.xml 2 1.0 1.0
```

5.2.4 Bose-Hubbard model generator `hamgen_B`

`hamgen_B` is a utility tool generating a hamiltonian file describing Bose-Hubbard model

```
$ hamgen_B [-o filename] M t V U F
```

The meaning of parameters is following:

`M` The maximum number of sites on a site

`t` The hopping parameter

`V` The nearest neighbor interaction. Positive for repulsive potential and negative for attractive.

`U` The onsite interaction. Positive for repulsive potential and negative for attractive.

`F` The chemical potential on a site per a bond connected to the site, $F = h/z$, where z is the coordination number, for example, $z = 4$ for the square lattice.

`filename` Name of the Hamiltonian file (default: `hamiltonian.xml`).

5.2.5 Algorithm file generator `dla_alg`

`dla_alg` is an utility tool to convert a hamiltonian file to an algorithm file.:

```
$ dla_alg HFILE AFILE
```

The meaning of parameters are following:

`HFILE` The Hamiltonian file to be loaded (default: `hamiltonian.xml`).

`AFILE` The algorithm file to be generated (default: `algorithm.xml`).

5.2.6 Structure factor file generator `sfgene`

`sfgene` is a utility tool generating a structure factor file for a hypercubic lattice:

```
$ sfgene [-o filename] D L_1 ... L_D Ntau Ntau_cutoff KTYPE
```

The meaning of parameters are following:

`D` Dimension of lattice

`L_1 ... L_D` Linear length of lattice in each dimension.

`Ntau` The number of discretization of imaginary time

`Ntau_cutoff` The maximum number of distance in imaginary time $d\tau$

`KTYPE` Pattern of wave vectors k

- `KTYPE==0`

  Wave vectors with $k_x = n\pi/L_x, n = 0, 2, \ldots, L$ are calculated (k_y and k_z are zero for all wave vectors).

- `KTYPE==1`

  $k/\pi = (0,0,0), (1,0,0), (0,1,0), (1,1,0), \ldots, (1,1,1)$ for three dimensional case.

`filename` The structure factor file to be generated (default: `sf.xml`).

5.2.7 Real space temperature Green's function file generator `cfgene`

`cfgene` is a utility tool generating a real space temperature Green's function file for a hypercubic lattice:

```
$ cfgene [-o filename] D L_1 ... L_D Ntau
```

The meaning of parameters is following:

`D` The dimension of lattice

`L_1 ... L_D` The linear lengths of lattice in each dimension.

`Ntau` The number of discretization of imaginary time

`filename` The real space temperature Green's function file to be generated (default: `sf.xml`).

5.3 Directed loop algorithm solvers, `dla_H` and `dla_B`

`dla_H` and `dla_B` are quantum Monte Carlo solvers with the directed loop algorithm. They take an input file as the command line argument. `dla_H` is for spin system and `dla_B` is for Bosonic system.

```
$ dla_H param.in
```

DSQSS/DLA implements random number parallelization by using MPI. Since each process performs `NSET` Monte Carlo calculation independently, the total number of MC sets increases N_{process} and thus it is expected that the statistical error also reduces to $1/\sqrt{N_{\text{process}}}$ times.

```
$ mpiexec -np 4 dla_B param.in
```

5.4 Output of DSQSS/DLA

5.4.1 Format

DSQSS/DLA generates the result of a simulation as a plain-text file. The first character stands for the meaning of the line.

`P <name> = <value>` Parameters read from the input files.

`R <name> = <mean> <error>` Results of observables. `<mean>` denotes the expected value and `<error>` denotes the statistical error of `<mean>`.

`I <text> = <value>` Other information.

`C <text>` Comments.

The following one is a result of an antiferromagnetic Heisenberg chain.

```
C This is DSQSS ver.1.2.0

P D       =            1
P L       =            8
P BETA    =      10.0000000000000000
P NSET    =           10
P NMCSE   =         1000
P NMCSD   =         1000
```

(continues on next page)

(continued from previous page)

```
P NMCS    =         1000
P SEED    =    198212240
P NSEGMAX =        10000
P NVERMAX =        10000
P NCYC    =            7
P ALGFILE = algorithm.xml
P LATFILE = lattice.xml
P CFINPFILE  = cf.xml
P SFINPFILE  = sf.xml
P CKINPFILE  = sf.xml
P OUTFILE    = res.dat.000
P CFOUTFILE  = cfout.dat.000
P SFOUTFILE  = sfout.dat.000
P CKOUTFILE  = ckout.dat.000
P SIMULATIONTIME   =     0.000000
R anv = 3.03805000e+00 1.25395375e-02
R ene = -4.55991910e-01 1.20267537e-03
R spe = -1.76672204e-02 4.09064489e-02
R len = 1.20014021e+01 4.78403202e-02
R xmx = 3.00035053e-01 1.19600800e-03
R amzu = -2.00000000e-04 1.08972474e-04
R bmzu = -2.00000000e-04 1.08972474e-04
R smzu = 1.32382500e-03 1.40792745e-04
R xmzu = 1.32382500e-02 1.40792745e-03
R amzs = -9.25000000e-04 4.02247160e-03
R bmzs = -2.03918502e-04 2.22828174e-03
R smzs = 8.72503175e-01 8.93939492e-03
R xmzs = 3.00500011e+00 2.99056535e-02
R time = 9.01378000e-08 1.61529255e-09
I [the maximum number of segments]          = 123
I [the maximum number of vertices]          = 66
I [the maximum number of reg. vertex info.] = 3
```

5.4.1.1 Notations

N_s The number of sites.

$Q(\vec{k})$ The Fourier transformation of an arbitrary operator on a site i, Q_i.

$$Q(\vec{k}) \equiv \frac{1}{\sqrt{N_s}} \sum_i^{N_s} Q_i e^{-i\vec{r}_i \cdot \vec{k}}$$

$Q(\tau)$ An arbitrary operator at imaginary time τ.

$$Q(\tau) \equiv \exp\left[\tau \mathcal{H}\right] Q(\tau = 0) \exp\left[-\tau \mathcal{H}\right]$$

$\tilde{Q}$ The average of an arbitrary operator Q over the imaginary time, $\frac{1}{\beta} \int_0^\beta \mathrm{d}\tau Q(\tau)$

M^z The component of a local degree of freedom along with the quantized axis. For example, z component of the local spin operator S^z for spin systems and the number operator n for the Bose-Hubbard models.

$M^\pm$ The ladder operator. $M^\pm \equiv S^\pm$ for spin systems, and the creation/annihilation operators $M^+ \equiv b^\dagger$, $M^- \equiv b$ for the Bose-Hubbard models.

M^x The off-diagonal order parameter. $M^x \equiv (S^+ + S^-)/2$ for spin systems and $M^x \equiv (b + b^\dagger)$ for the Bose-Hubbard models.

T The temperature.

β The inverse temperature.

h The conjugate field to the operator M^z . The longitudinal magnetic field for spin systems and the chemical potential for the Bose-Hubbard models.

$\langle Q \rangle$ The expectation value of an arbitrary operator Q over the grand canonical ensemble.

5.4.2 Main results

Main results are written in a file with the name specified by `outfile` keyword in the input parameter file.

`sign` The sign of the weights.

$\sum_i W_i / \sum_i |W_i|$

`anv` The mean number of the vertices.

$\frac{\langle N_v \rangle}{N_s}$

`ene` The energy density (energy per site)

$\epsilon \equiv \frac{1}{N_s} \left(E_0 - T \langle N_v \rangle \right)$

`spe` The specific heat

$C_V \equiv \frac{\partial \epsilon}{\partial T}$

`len` The mean length of worm

`xmx` The transverse susceptibility

`amzu` The "magnetization" (uniform, $\tau = 0$).

$\langle m^z \rangle$, where $m^z \equiv \frac{1}{N_s} \sum_i^{N_s} M_i^z$

`bmzu` The "magnetization" (uniform, average over τ). $\langle \tilde{m}^z \rangle$.

`smzu` The structure factor (uniform).

$$S^{zz}(\vec{k}=0) \equiv \frac{1}{N_s} \sum_{i,j} e^{i\vec{k}\cdot(\vec{r}_i - \vec{r}_j)} \left[\left\langle M_i^z M_j^z \right\rangle - \left\langle M_i^z \right\rangle \left\langle M_j^z \right\rangle \right] \Bigg|_{\vec{k}=0} = N_s \left[\left\langle (m^z)^2 \right\rangle - \left\langle m^z \right\rangle^2 \right]$$

`xmzu` The longitudinal susceptibility (uniform).

$$\chi^{zz}(\vec{k}=0, \omega=0) \equiv \frac{\partial \left\langle \tilde{m}^z \right\rangle}{\partial h} = \beta N_s \left[\left\langle (\tilde{m}^z)^2 \right\rangle - \left\langle \tilde{m}^z \right\rangle^2 \right]$$

`amzs` The "magnetization" ("staggered", $\tau = 0$)

$\langle m_s^z \rangle$, where $m_s^z \equiv \frac{1}{N_s} \sum_i^{N_s} M_i^z \cos\left(2\pi \frac{\text{mtype}(i)}{N_{\text{mtype}}} \right)$, mtype(i) is the kind of measurement of i site (see lattice file), and N_{mtype} is the number of kinds of measurements.

`bmzu` The "magnetization" ("staggered", average over τ). $\langle \tilde{m}_s^z \rangle$.

`smzs` The structure factor ("staggered").

$$S^{zz}(\vec{k}_s) = N_s \left[\left\langle (m_s^z)^2 \right\rangle - \left\langle m_s^z \right\rangle^2 \right]$$

xmzs The longitudinal susceptibility ("staggered").

$$\chi^{zz}(\vec{k}_s, \omega = 0) = \beta N_s \left[\left\langle (\tilde{m}_s^z)^2 \right\rangle - \left\langle \tilde{m}_s^z \right\rangle^2 \right]$$

5.4.3 Structure factor output

The structure factor is written into a file with the name specified by `sfoutfile` keyword in the input file. The structure factor is defined as the following:

$$S^{zz}(\vec{k}, \tau) \equiv \left\langle M^z(\vec{k}, \tau) M^z(-\vec{k}, 0) \right\rangle - \left\langle M^z(\vec{k}, \tau) \right\rangle \left\langle M^z(-\vec{k}, 0) \right\rangle$$

Wave vector $\vec{k}$ and imaginary time τ are specified by the name `C<k>t<t>` as the following:

```
R C0t0 = 1.32500000e-03 1.40929454e-04
R C0t1 = 1.32500000e-03 1.40929454e-04
R C1t0 = 7.35281032e-02 3.18028565e-04
```

where `<k>` is an index of the wave vector specified by `kindex` (the last element of each `SF` tag) in the structure factor input file and `<t>` is an index of the discretized imaginary time.

5.4.4 Real space temperature Green's function output

The real space temperature Green's function is written into a file with the name specified by `cfoutfile` keyword in the input file. The real space temperature Green's function is defined as the following:

$$G(\vec{r}_{ij}, \tau) \equiv \left\langle M_i^+(\tau) M_j^- \right\rangle$$

Displacement $\vec{r}_{ij}$ and imaginary time τ are specified by the name `C<k>t<t>` as the same way of structure factor, where `<k>` is an index of the displacement specified by `kind` (the first element of each `CF` tag) in the real space temperature Green's function input file, and `<t>` is an index of the discretized imaginary time.

5.4.5 Momentum space temperature Green's function output

The momentum space temperature Green's function is written into a file with the name specified by `ckoutfile` keyword in the input file. The momentum space temperature Green's function is defined as the following:

$$G(\vec{k}, \tau) \equiv \left\langle M^+(\vec{k}, \tau) M^-(-\vec{k}, 0) \right\rangle$$

Wave vector $\vec{r}_{ij}$ and imaginary time τ are specified by the name `C<k>t<t>` as the same way of structure factor, where `<k>` is an index of the displacement specified by `kind` (the last element of each `SF` tag) in the momentum space temperature Green's function input file, and `<t>` is an index of the discretized imaginary time.

CHAPTER

SIX

TUTORIAL OF DSQSS/PMWA

6.1 About DSQSS/PMWA

`DSQSS / PMWA` is a numerical package of the world-line quantum Monte Carlo method with non-local updates algorithm for large-scale parallel computing. PMWA can treat $S = 1/2$ quantum many-body system such as XXZ spin model, Heisenberg model, transverse Ising model model and hard-core boson system with large lattice sizes at finite temperatures. Basically, any physical quantities can be calculated within the statistical error. By default, the energy under a longitudinal worm source field, longitudinal magnetization, transverse magnetization, winding number, and imaginary time $S^z S^z$ spin correlation function are calculated.

6.2 Usage

PMWA can be used by the following procedures:

1. Make an input file for a lattice (use `lattgene_P`).
2. Make an input file.
3. Execute PMWA (boson system:`pmwa_B` , spin system:`pmwa_H`)
4. Generate output files.

There is also a simple tool `pmwa_pre` for executing 1 and 2 at once. In this section, the usage and parameters are explained for each procedure.

6.2.1 Make an input file for a lattice

In PMWA, as a simple tool `lattgene` is prepared for generating a lattice definition file `lattice.xml` for a standard model. In this section, the usage of `lattgene` is explained.

`lattgene` generates a lattice definition file for a cubic lattice. The parameters to be specified are as follows.

Parameter	type	Remarks
D	int	The dimensions
L	int	The size of the lattice (specified for each dimension consecutively)
B	double	The inverse temperature
NLdiv	int	The number of divisions of L (divide NLdiv for each dimension)
NBdiv	int	The number of divisions of L
NFIELD	int	The type of magnetic field (basically set to 0)

Examples of use are described below.

1. One-dimensional 8-sites chain, inverse temperature 10.0, definition of lattice file when division number is 1:

   ```
   $ lattgene 1 8 10.0 1 1 0
   ```

2. Two dimensional 4*4 sites lattice, reverse temperature 10.0, definition of lattice file when division number is 1:

   ```
   $ lattgene 2 4 4 10.0 1 1 0
   ```

6.2.2 Make an input file for DSQSS/PMWA

To execute PMWA, a text format input file is required. There are two kinds of parameters in the input file: parameters specifying calculation conditions and parameters specifying a Hamiltonian.

An example of the input file is shown below:

```
RUNTYPE = 0
NSET    = 10
NMCS    = 10000
NPRE    = 10000
NTHERM = 10000
NDECOR = 10000
SEED   = 31415
NC     = 0
NVERMAX = 10000000
NWORMAX = 1000
latfile    = lattice.xml
outfile    = sample.log
CB       = 2
G        = 0.3
U        = 0
V        = 3
t        = 1
MU       = 2
NMAX     = 1
```

The meaning of each parameter is as follows.

- Parameter of calculation conditions

Parameter	type	Remarks
RUNTYPE	int	Calculation mode (0: normal calculation, 1: restart calculation)
CB	int	Initial arrangement (0: Vacuum, 1: Checker-Board, 2: Random)
NSET	int	The repetition number of Monte Carlo calculation
NMCS	int	The number of Monte Carlo sweeps used for physical quantity calculation
NPRE	int	The number of Monte Carlo steps used for precalculation to determine the number of trials to generate worms per sweep
NMCSE	int	The number of Monte Carlo sweeps used for initial relaxation
NMCSD	int	The number of Monte Carlo sweeps between measurements
SEED	int	If it is 0 or more, seed is actually used, if negative, seed is given with random number.
NVERMAX	int	The maximum number of vertexes (default is 10^8). There is no upper limit when -1.
NWORMAX	int	The maximum number of worms (default is 10^3). There is no upper limit when -1.
SFINPFILE	str	If this file name is given, calculate structure factors specified in this file.
SFOUTFILE	str	If SFINPFILE is given, output structure factors to the specified file (default is sf.out).
Step_x	int	Give the spatial width when computing the correlation function (default: 1).
Step_k	int	Give the width of wavenumber when computing correlation function of wavenumber representation (default: 1).
CFOUTFILE	str	When it is input, it outputs the correlation function to the specified file (default is cf.out).

Here, NVERMAX and NWORMAX are automatically resized and determined (implemented in ver. 1.2.0).

- Parameter of interactions

PMWA can treat the models for hard-core boson and $S = 1/2$ XXZ system. The Hamiltonian for hard-core boson system is given by

$$\mathcal{H} = -t_b \sum_{\langle i,j \rangle} b_i^\dagger b_j + U_{BB} \sum_i n_i(n_i - 1) + V_{B1} \sum_{\langle i,j \rangle} n_i n_j + \mu \sum_i n_i - \Gamma \sum_i (b_i^\dagger + b_i),$$

where $\langle i, j \rangle$ indicates a nearest-neighbor pair. For $S = 1/2$ XXZ model, the hamiltonian is given by

$$\mathcal{H}^{XXZ} = -J_{xy} \sum_{\langle i,j \rangle} (S_i^x S_j^x + S_i^y S_j^y) - J_z \sum_{\langle i,j \rangle} S_i^z S_j^z - H \sum_i S_i^z - \Gamma \sum_i S_i^x.$$

The parameters specified in the input file and the parameters in the above expression correspond as follows.

Parameter	Boson	Spin
t	t_b	J_{xy}
U	U_{BB}	-
V	V_{B1}	J_z
MU	μ	H
G	Γ	$\Gamma/2$

Each parameter is specified as double type.

6.2.3 Run DSQSS/PMWA

After generating the input file, PMWA runs by typing the following command (the name of the input file is assumed as `param.in`). Execution file is different depending on the system, i.e. the spin system or the hard core boson system.

1. Spin system

   ```
   $pmwa_H param.in
   ```

2. Hard core boson system

   ```
   $pmwa_B param.in
   ```

After calculation, one result file and two temporary files for restart mode (evout_sample.log, RND_evout_sample.log) are outputted.

6.2.4 Output files

Here, the parameters specific to PMWA for one result file and two temporary files for restart mode are described.

- Result file

Type	Name	Description
P	L	The size of the lattice.
P	DOML	The size of the domain divided by parallelization
P	DOMBETA	The size of the inverse temperature divided by parallelization
P	NDIVL	The number of lattice divisions
P	NTEST	Number of samples to be tested (for details, see the description of Monte Carlo calculation)
R	nver	The number of kinks and worms
R	nkin	The number of kinks
R	wndx	The expected value of square of winding number in x direction
R	wndy	The expected value of square of winding number in y direction
R	wndz	The expected value of square of winding number in z direction
R	wnd2	The total number of squares of winding number(wndx+wndy+wndz)
R	bmxu	The expected value of S_x (uniform τ integral)
R	bmpu	The expected value of S_+ (uniform τ integral)
R	bmmu	The expected value of S_- (uniform τ integral)
R	comp	The compressibility
R	lxmx	The local worm number fluctuation at each site
I	the maximum number of vertices	The maximum number of vertices
I	the maximum number of worms	The maximum number of worms

Here, the type is the letter given to the beginning of each line of output. P, R, I indicate Parameter, Result, Information respectively.

- Output files for restart mode

PMWA implements the restart function, and if there are two files below, a restart is automatically performed. Below, a brief description of the output contents of each file is described.

1. evout_sample.log

   Files outputted for the number of cycles at the end of computation, world lines information, and vertex information. For restart calculation, calculations are performed with the loaded arrangement as the initial condition.

```
26 : Number of cycles at the end of calculation
0 1 : Information on the world line at site :math:`0` in the domain.
i/N beta, (i+1)/N beta Information on the world line of the section : 0:␣
↪down, 1: up
0 0 :Information on the world line 1 at the site 1 in the domain.
1 1  :Information on the world line 2 at the site 2 in the domain.
...
8 0.056997107 2 1 4  Vertex label, tau, type of vertex, number of world␣
↪lines, bond number
9 0.056997107 2 0 5
44 0.28066013 2 1 3
```

Here, the types of vertexes are defined as follows.

The type of vertex	Description
-5	The start and end points in the imaginary time direction in each domain. It does not need to split domain.
-4(right), -2(left)	Vertex diagonally across domains. It does not need to split domain.
-3(right), -1(left)	Vertex off-diagonally across domains. It does not need to split domain.
0	On site vertex (except worms).
1	Two site vertex
2	Kink.
3	Two site vertex (next nearest) (it is left for compatibility).
4	The worm moving at that time (annihilation operator).
5	The worm stopping at that time (creation operator).
6	The worm stopping at that time (not related to neither annihilation or creation operators) or needless vertex.
7	Marker(for imaginary time correlation function measurement).

2. RNDevout_sample.log

   The output binary file for objects generating random numbers. At recalculation, calculations are performed with the random number information which is read as the initial condition.

6.3 Calculation of the energy on spin chain by DSQSS/PMWA

In this tutorial, you learn how to use DSQSS / PMWA by calculating the ground state energy of antiferromagnetic Heisenberg chain of $S = 1/2$.

The calculation by DSQSS / PMWA is done in the following three stages:

1. Make an input file,
2. Perform calculation,
3. Analyze calculation results.

6.3.1 Make an input file

To run DSQSS/PMWA, following two input files are needed:

1. lattice definition file,
2. parameter file.

The utility tool `pmwa_pre` for doing automatically these process is prepared. This tool is made by a Python script that generates input files for DSQSS/DLA and DSQSS/PMWA from a single input file. First of all, prepare a text file `std.in` with the following contents as an input file for `pmwa_pre`.

```
[System]
solver = PMWA
[Hamiltonian]
model_type = spin
Jxy = -1.0
Jz = -1.0
Gamma = 0.1
[Lattice]
lattice_type = square
D = 1
L = 16
Beta = 100
[Parameter]
CB = 1
SEED = 31415
NSET = 10
NMCS = 100
NPRE = 100
NTHERM = 100
NDECOR = 100
```

This file is located in `sample/pmwa/1DDimer` directory. Next, type the following command:

```
$ $DSQSS_ROOT/bin/pmwa_pre std.in
```

Then, input files `param.in` and `lattice.xml` are generated.

6.3.2 Perform calculation

After making input files, the calculation by `DSQSS/PMWA` starts by typing the following command:

```
$ $DSQSS_ROOT/bin/pmwa_H param.in
```

Random numbers parallel computation is possible by using MPI (it is also possible to perform parallel computation by dividing space/imaginary time. Please see DLA user manual for details).

```
$ mpiexec -np 4 $DSQSS_ROOT/bin/pmwa_H param.in
```

Calculation can be done independently for parallel number (4 for the above example), and the accuracy can be improved by increasing the number of Monte Carlo samples.

6.3.3 Analyze calculation results

The calculation result is written to the output file `sample.log`. The energy is obtained by typing the following command:

```
$ grep ene sample.log
R ene  = -0.5705441 0.0003774399737579577
```

In the case of DSQSS / PMWA, it is necessary to insert a transverse magnetic field. Therefore, please note that extrapolation is necessary to obtain the result at zero magnetic field.

CHAPTER

SEVEN

USER'S MANUAL FOR DSQSS/PMWA

7.1 Input files for DSQSS/PMWA

In the input file of DSQSS/DLA and DSQSS/PMWA, there are many common parameters. In this section, the parameters which are not included in DSQSS/DLA or where the usage is different from DSQSS/DLA are explained.

- Parameters for calculation conditions

Parameter	type	default	description
RUNTYPE	int		Calculation mode (0: from scratch, 1: restart)
CB	int	0	Initial pattern (0: Vacuum, 1: Checker-Board, 2: Random)
NWORMAX	int	1000	The maximum number of worms. When the value is equal to be -1, the upper limit becomes infinity.
Step_x	int	1	The spatial width in calculating the correlation functions.
Step_k	int	1	The width of the wave-number space in calculating correlation functions of wave-number representation.

- Parameters related to the model

Parameter	type	description
t	double	t in boson system. J_{xy} in spin system.
U	double	U in boson system. This parameter is not used in spin system.
V	double	V in boson system. J_z in spin system.
MU	double	μ in boson system. H in spin system.
G	double	Γ in boson system. $\Gamma/2$ in spin system.
NMAX	-	This value is not used (always fixed to be 1).

An example of the input file is shown as follows:

```
RUNTYPE = 0
NSET  = 10
NMCS  = 1000
NPRE = 1000
NTHERM = 1000
NDECOR = 1000
SEED  = 31415
NC    = 0
NVERMAX = 10000000
NWORMAX = 1000
algfile   = algorithm.xml
latfile   = lattice.xml
outfile   = sample.log
CB      = 2
G       = 0.3
U       = 0
V       = 3
t       = 1
MU      = 2
NMAX    = 1
```

7.2 Output files for DSQSS/PMWA

DSQSS/PMWA outputs one result file and two temporary files for restart calculation (evout_sample.log, RND_evout_sample.log).

- Result file

  Many parameters are common to DSQSS/DLA. In this section, the parameters which are not included in DSQSS/DLA or

Type	Name	Description
P	L	The size of the lattice.
P	DOML	The size of the domain divided by parallelization
P	DOMBETA	The size of the inverse temperature divided by parallelization
P	NDIVL	The number of lattice divisions
P	NTEST	Number of samples to be tested (for details, see the description of Monte Carlo calculation)
R	nver	The number of kinks and worms
R	nkin	The number of kinks
R	wndx	The expected value of square of winding number in x direction
R	wndy	The expected value of square of winding number in y direction
R	wndz	The expected value of square of winding number in z direction
R	wnd2	The total number of squares of winding number(wndx+wndy+wndz)
R	bmxu	The expected value of S_x (uniform τ integral)
R	bmpu	The expected value of S_+ (uniform τ integral)
R	bmmu	The expected value of S_- (uniform τ integral)
R	comp	The compressibility
R	lxmx	The local worm number fluctuation at each site
I	the maximum number of vertices	The maximum number of vertices
I	the maximum number of worms	The maximum number of worms

Here, the type means the letter given to the beginning of each line of output. P, R, I indicate Parameter, Result, Information, respectively.

- File for restart calculation

  PMWA implements the restart function, and if there are two files `evout_sample.log` and `RNDevout_sample.log`, a restart is forcibly performed. In the following, the output contents of each file are briefly described.

  1. evout_sample.log

  In this file, the number of cycles at the end of computation, information on world lines, and vertex information are outputted. In restart calculation, the loaded arrangement is set as the starting condition.

```
26 : Number of cycles at the end of calculation.
0 1 : Information on the world line at 0-th site in the domain.
Information on the world line of the section between i/N beta, (i+1)/N beta; 0:␣
↪down, 1: up
0 0 : Information on the world line at the 1st site in the domain.
1 1 : Information on the world line at the 2nd site in the domain.
...
8 0.056997107 2 1 4 :lavel for vertex, tau, kind of vertex, Number of world lines,
↪ Bond number
9 0.056997107 2 0 5
44 0.28066013 2 1 3
```

PMWA implements the restart function, and if there are two files below, a restart is automatically performed. Below, a brief description of the output contents of each file is described.

1. evout_sample.log

Files outputted for the number of cycles at the end of computation, world lines information, and vertex information. For restart calculation, calculations are performed with the loaded arrangement as the initial condition.

```
26 : Number of cycles at the end of calculation
0 1 : Information on the world line at site :math:`0` in the domain.
i/N beta, (i+1)/N beta Information on the world line of the section : 0:␣
↪down, 1: up
0 0 :Information on the world line 1 at the site 1 in the domain.
1 1  :Information on the world line 2 at the site 2 in the domain.
...
8 0.056997107 2 1 4  Vertex label, tau, type of vertex, number of world␣
↪lines, bond number
9 0.056997107 2 0 5
44 0.28066013 2 1 3
```

Here, the types of vertexes are defined as follows.

The type of vertex	Description
-5	The start and end points in the imaginary time direction in each domain. It does not need to split domain.
-4(right), -2(left)	Vertex diagonally across domains. It does not need to split domain.
-3(right), -1(left)	Vertex off-diagonally across domains. It does not need to split domain.
0	On site vertex (except worms).
1	Two site vertex
2	Kink.
3	Two site vertex (next nearest) (it is left for compatibility).
4	The worm moving at that time (annihilation operator).
5	The worm stopping at that time (creation operator).
6	The worm stopping at that time (not related to neither annihilation or creation operators) or needless vertex.
7	Marker(for imaginary time correlation function measurement).

2. RNDevout_sample.log

   The output binary file for objects generating random numbers. At recalculation, calculations are performed with the random number information which is read as the initial condition.