Documentation

Archives and Files

The parent directory *projects/* contains the following archives:

- *autocorrelation_analysis/*
  Programs from Hans-Gerd Evertz to analyze autocorrelation time and binning series. His lecture notes on computer simulations are also here.

- *spin_Z3/*
  $\mathbb{Z}_3$ spin model programs and papers.

  - *papers/*
    All published papers.
  - *metropolis_conventional_rep/*
    Fortran programs to simulate the conventional representation of the model (by Christof).
  - *small_lattice_exact/*
    Fortran programs to compute the exact value of the observables in the conventional representation, valid only for very small lattices, $< 4^4$ (by Christof).
  - *worm_z3/*
    C/C++ programs to simulate the dual representation of the model using the worm algorithm.

- *spin_SU3/*
  $SU(3)$ spin model programs and papers.

  - *papers/*
    All published papers.
  - *conventional_rep/*
    C/C++ programs to simulate the conventional representation of the model.
  - *dual_rep/*
    C/C++ programs to simulate the dual representation of the model using local updates. The worm algorithm did not work for this model. There are two different ways of writing the dual variables (see papers):

    - *metropolis_original_var/*
      Original variables, with 2 constrained link variables and 2 constrained monomers.
    - *metropolis_rotated_var/*
      Rotated variables. The update with the new dimers and monomers has a better performance.

- *gauge_Higgs_model/*
  All gauge Higgs models programs and papers.

- *papers/*
  All published papers.
- *Z3_model/*
  C/C++ programs to simulate the dual representation of the $\mathbb{Z}_3$ gauge Higgs model using the surface worm algorithm.
- *U1_model/*
  C/C++ programs to simulate the dual representation of the $U(1)$ gauge Higgs model (with only one flavor) using the surface worm algorithm.
- *U1_model_2flavors/*
  C/C++ programs to simulate the dual representation of the $U(1)$ gauge Higgs model (two scalar fields) using the surface worm algorithm.

File names

- Main programs:
  - *gen.cpp*
    Main program to generate the configurations. The output file is in binary format.

    Warning!! By configurations we mean the total number of occupation numbers needed to compute the observables, i.e. we do not store the whole lattice.
  - *analysis.cpp*
    Main analysis program to compute the mean value and error bars of the observables. It reads the configuration files created by *gen.cpp* and creates a file with the suffix *obs*, where all observables are printed in text format.
- Description of the include files:
  - *gen.h*: Libraries and global variables.
  - *lattice.h*: Global lattice variables and initialization routine of the neighbors array.
  - *init.h*: Initialization routines
  - *weights.h*: Functions to precompute the weights used in the accept/reject step.
  - *worm.h*: Worm algorithm.
  - *sweeps.h*: Sweeps with local updates.
  - *multigrid.h*: Multigrid equilibration subroutine.
  - *ranlxd.**: Lüscher's random number generator in double precision.
- *makefile*
  To compile the programs for different parameters and lattice sizes.

Simulation of the $\mathbb{Z}_3$ spin model

- To compile the configuration files:

```
cd worm_z3/
make ran
make SIZE=4 PAR=TAU gen
```

  - The variable `SIZE` is equal to the lattice size.
  - The variable `PAR` can be equal to `TAU, KAPPA` or `MU`. The configurations are generated as a function of the variable `PAR`.
  - The executable file is created in *bin/*

- To generate the configurations:

```
cd worm_z3/bin
./gen$(SIZE)_$(PAR).x
```

  - The input parameters are located in *bin/worm_par.start*, where par is mu, tau or kappa.
  - The executable `gen$(SIZE)_$(PAR).x` reads the file with the input parameters and creates a binary file with the configurations.

- To compile the analysis program:

```
cd worm_z3/
make PAR=TAU anal
```

  - The variable `PAR` can be equal to `TAU, KAPPA` or `MU`. The observables are function of the variable `PAR`.
  - The executable file is created in *bin/*

- To analyze the configurations:

```
cd worm_z3/bin
./anal_$(PAR).x -f CONFIGURATION_FILE
```

  - The input parameter is the name of the file where the configurations are stored (same name from *bin/worm_par.start*).
  - The executable `anal_$(PAR).x` reads the `CONFIGURATION_FILE` file and creates a text file `CONFIGURATION_FILEobs` with the observables and error bars.

Simulation of the SU(3) spin model

The compilation and execution of programs is exactly the same for both rotated and original variables. Here I will use the original variables to show the steps.

- To compile the configuration files:

```
cd dual_rep/metropolis_original_var/
make ran
make SIZE=4 PAR=TAU gen
```

  - The variable `SIZE` is equal to the lattice size.
  - The variable `PAR` can be equal to `TAU, KAPPA0` or `MU`.
  - For `TAU` and `MU`, the configurations are generated as a function of the variable `PAR`.
  - `KAPPA0` generates configurations as a function of β for $\kappa = 0$.
  - The executable file is created in *bin/*

- To generate the configurations:

  ```
  cd dual_rep/metropolis_original_var/bin
  ./gen$(SIZE)_$(PAR).x
  ```

  - The input parameters are located in *bin/metro_su3_par.start*, where par is tau or mu.
  - The executable `gen$(SIZE)_$(PAR).x` reads the file with the input parameters and creates a binary file with the configurations.

- To compile the analysis program:

  ```
  cd dual_rep/metropolis_original_var/
  make PAR=TAU anal
  ```

  - The variable `PAR` can be equal to `TAU` or `MU`. The observables are function of the variable `PAR`.
  - The executable file is created in *bin/*

- To analyze the configurations:

  ```
  cd dual_rep/metropolis_original_var/bin
  ./anal_$(PAR).x -f CONFIGURATION_FILE
  ```

  - The input parameter is the name of the file where the configurations are stored (same name from *bin/metro_su3_par.start*).
  - The executable `anal_$(PAR).x` reads the `CONFIGURATION_FILE` file and creates a text file `CONFIGURATION_FILEobs` with the observables and error bars.

Simulation of the $\mathbb{Z}_3$ gauge-Higgs model

- To compile the configuration files:

  ```
  cd gauge_Higgs_model/Z3_model/
  make ran
  make NS=4 NT=4 PAR=BETA gen
  ```

  - The variable `NS` is the length of the spatial direction of the lattice.
  - The variable `NT` is the length of the temporal direction of the lattice.
  - The variable `PAR` can be equal to `BETA` or `MU`. The configurations are generated as a function of the variable `PAR`.
  - The executable file is created in *bin/*

- To generate the configurations:

  ```
  cd gauge_Higgs_model/Z3_model/bin
  ./gen$(NS)x$(NT)_$(PAR).x
  ```

  - The input parameters are located in *bin/worm_par.start*, where par is beta or mu.
  - The executable `gen$(NS)x$(NT)_$(PAR).x` reads the file with the input parameters and creates a binary file with the configurations.

- To compile the analysis program:

  ```
  cd gauge_Higgs_model/Z3_model/
  make PAR=BETA anal
  ```

  - The variable `PAR` can be equal to `BETA` or `MU`. The observables are function of the variable `PAR`.
  - The executable file is created in *bin/*

- To analyze the configurations:

  ```
  cd gauge_Higgs_model/Z3_model/bin
  ./anal_$(PAR).x -f CONFIGURATION_FILE
  ```

  - The input parameter is the name of the file where the configurations are stored (same name from *bin/worm_par.start*).
  - The executable `anal_$(PAR).x` reads the `CONFIGURATION_FILE` file and creates a text file `CONFIGURATION_FILEobs` with the observables and error bars.

Simulation of the U(1) gauge-Higgs model

- To compile the configuration files:

  ```
  cd gauge_Higgs_model/U1_model/
  make ran
  make SIZE=4 gen
  ```

  - The variable `SIZE` is the size of the lattice.
  - The configurations are generated as a function of β.
  - The executable file is created in *bin/*
  - External libraries: GSL and BLAS.

- To generate the configurations:

  ```
  cd gauge_Higgs_model/U1_model/bin
  ./gen$(SIZE).x
  ```

  - The input parameters are located in *bin/worm_beta.start*.
  - The executable `gen$(SIZE).x` reads the file with the input parameters and creates a binary file with the configurations.

- To compile the analysis program:

  ```
  cd gauge_Higgs_model/U1_model/
  make anal
  ```

  - The observables are function of the variable β.
  - The executable file is created in *bin/*

- To analyze the configurations:

  ```
  cd gauge_Higgs_model/U1_model/bin
  ./anal.x -f CONFIGURATION_FILE
  ```

  - The input parameter is the name of the file where the configurations are stored (same name from *bin/worm_beta.start*).
  - The executable `anal.x` reads the `CONFIGURATION_FILE` file and creates a text file `CONFIGURATION_FILEobs` with the observables and error bars.

Simulation of the U(1) gauge-Higgs model with two scalar fields

- To compile the configuration files:

  ```
  cd gauge_Higgs_model/U1_model_2flavors/
  make ran
  make NS=4 NT=4 gen
  ```

  - The variable `NS` is the length of the spatial direction of the lattice.
  - The variable `NT` is the length of the temporal direction of the lattice.
  - The executable file is created in *bin/*
  - External libraries: GSL, BLAS and the include files of the BOOST library.

- To generate the configurations:

  ```
  cd gauge_Higgs_model/U1_model_2flavors/bin
  mpiexec -np 2 ./gen$(NS)x$(NT).x [OPTION] ...  [OUTPUT_FILE]
  ```

  - `gen$(NS)x$(NT).x` is an `MPI` program. The number of processes is equal to the number of points in the parameter space.
  - The executable `gen$(NS)x$(NT).x` reads the input parameters and creates a binary file per process with the total value of occupation numbers. The output files are named: `OUTPUT_FILE_PROCESSID.out`.
  - The input parameters are:
    - *-l*: lambda.
    - *-K*: initial value of κ.
    - *-k*: step size for κ.
    - *-B*: initial value of β.
    - *-b*: step size for β.
    - *-M*: initial value of μ.

    * *-m*: step size for μ.
    * *-n*: number of measurements.
    * *-s*: discarded steps between measurements.
    * *-e*: equilibration steps.
    * *-r*: Read initial configuration from `INPUT_FILE` file (for hot start).
    * *-h*: Flag. Print options.
    * *OUTPUT_FILE*: name of the output file (without extension).
- After the program has finished running, a file with the state of the last full configuration (values of occupation number per variable) will be written in `OUTPUT_FILE_PROCESSID.out.conf`

- **Example 1:** Generate data for 4 values of μ, starting at $\mu = 1$ with steps of $\Delta\mu = 0.1$. For $\lambda = 1$, $\beta = 1$, $\kappa = 5$, on a lattice of size $12^4 \times 4$. With 1000 measurements, 1000 equilibration steps and 10 skip steps. With cold start.

  ```
  cd gauge_Higgs_model/U1_model_2flavors/bin
  mpiexec -np 4 ./gen12x4.x -l 1 -B 1 -b 0 -K 5 -k 0 -M 1 -m 0.1
  -n 1000 -e 1000 -s 10 outfile
  ```

  The output files are:
  outfile_0.out for: $\lambda = 1$, $\beta = 1$, $\kappa = 5$, $\mu = 1$
  outfile_0.out.conf : file with last full configuration.
  outfile_1.out for: $\lambda = 1$, $\beta = 1$, $\kappa = 5$, $\mu = 1.1$
  outfile_1.out.conf : file with last full configuration.
  outfile_2.out for: $\lambda = 1$, $\beta = 1$, $\kappa = 5$, $\mu = 1.2$
  outfile_2.out.conf : file with last full configuration.
  outfile_3.out for: $\lambda = 1$, $\beta = 1$, $\kappa = 5$, $\mu = 1.3$
  outfile_3.out.conf : file with last full configuration.

- **Example 2:** Generate data for 2 values of β, starting at $\beta = 0.8$ with steps of $\Delta\beta = 0.25$. For $\lambda = 0.5$, $\mu = 0$, $\kappa = 3$, on a lattice of size $8^4 \times 4$. With 1000 measurements, 100 equilibration steps and 10 skip steps. With hot start, initial configuration stored in `initialconfig.out`.

  ```
  cd gauge_Higgs_model/U1_model_2flavors/bin
  mpiexec -np 2 ./gen8x4.x -l 0.5 -B 0.8 -b 0.25 -K 3 -k 0 -M 0 -m
  0 -n 1000 -e 100 -s 10 -r initialconfig.out outfile2
  ```
  The output files are:
  outfile2_0.out for: $\lambda = 0.5$, $\beta = 0.8$, $\kappa = 3$, $\mu = 0$
  outfil2e_0.out.conf : file with last full configuration.
  outfile2_1.out for: $\lambda = 0.5$, $\beta = 0.825$, $\kappa = 3$, $\mu = 0$
  outfile2_0.out.conf : file with last full configuration.

- To compile the analysis program:

  ```
  cd gauge_Higgs_model/U1_model_2flavors/
  make anal
  ```

  – The executable file is created in *bin/*

- To analyze the configurations:

  ```
  cd gauge_Higgs_model/U1_model_2flavors/bin
  ./anal.x -f CONFIGURATION_FILE
  ```

  – The input parameter is the name of the file where the configurations are stored (`CONFIGURATION_FILE = OUTPUT_FILE`).

  – The executable `anal.x` reads all the configuration files created by the generation program and writes the observables and error bars in the text file `CONFIGURATION_FILE.obs`.

  – The analysis program can read any number of measurements (not only the value given in the input parameters). For example, if the job stops in the middle of the generation of configurations, then the analysis program will take only the finished measurements.

BUGS/COMMENTS

Contact: Ydalia Delgado (ydelgado83@gmail.com)