

# ECMC-QCB

Yacob Ozdalkiran

Feb. 19, 2026

## 1. Introduction

This package written in C++ implements the generation of pure gauge configurations for lattice QCD with Event-Chain, Heatbath or Metropolis algorithms. Parallelisation is natively implemented with MPI and follows the checkboard (CB) pattern with frozen links.

Available observables to compute are the mean plaquette, the topological charge and the energy density. The last two are computed using the Wilson Flow with specified parameters and Lüscher's fast  $\mathfrak{su}(3)$  exponential.

## 2. Compilation

The code is compiled using CMake to generate the makefiles. A script `recompile.sh` is provided with the corresponding `bash` commands (compilation on the RUCHE cluster with module load).

To compile in general, place yourself at the root ECMC-QCB. The compilation commands are (replace [1] and [2] with your C and C++ MPI compilers) :

```
1 rm -rf build
2 mkdir build
3 cd build
4 cmake .. -DCMAKE_C_COMPILER=[1] -DCMAKE_CXX_COMPILER=[2]
5 make -j $(nproc)
6 cd ..
```

## 3. Usage

To execute a run, you need an input file. The templates of the input files for Heatbath and ECMC are available in the ECMC-QCB/inputs folder. You can run the simulation with the following command (replace [1] with `hb` for Heatbath or `ecmc` for ECMC) :

```
1 mpirun -np 16 ./build/gauge_[1]_cb inputs/[1]_cb
```

Notice that in the example inputs `n_core_dims=2`, which means that there are 2 cores per dimensions, or  $2^4 = 16$  cores in total. The number of cores you run the binary with (via the `-np` option on `mpirun`) should always be consistent with the `n_core_dims` value of the input file.

## 4. Input file

The input file can be split in six categories.

#### 4.1. Lattice parameters

- `L_core` : Number of sites per dimension of the squared sub-lattice contained in each MPI core. If the simulation is the continuation of an existing run, this parameter should be the same as the initial run.
- `n_core_dims` : Number of MPI core per dimensions. The total number of MPI cores is `n_core_dims`<sup>4</sup>. If the simulation is the continuation of an existing run, this parameter should be the same as the initial run.
- `cold_start` : `true` if the simulation starts cold, `false` for a hot start. Ignored if the run is the continuation of an existing run.

#### 4.2. Run parameters

- `seed` : Integer seed of the random number generator. If the simulation is the continuation of an existing run, this parameter is ignored (the state of the RNG of the last run for each MPI core is used instead).
- `N_shift_therm` : Number of thermalization random shifts. During these shifts, no observable is saved but the plaquette can be computed and accessed in the standard output.
- `N_shift` : Number of random shifts after the thermalization phase.
- `N_switch_eo` : Number of even/odd updates between two shifts.

#### 4.3. Algorithm parameters

- `beta` :  $\beta$  of the Wilson action.

##### 4.3.1. Heatbath

- `N_sweeps` : Number of Heatbath sweep during an even/odd update.
- `N_hits` : Number of Heatbath hit during a sweep.

##### 4.3.2. ECMC

- `param_theta_sample` : Total displacement angle of the chain during an even/odd update.
- `param_theta_refresh` : Total displacement angle of the chain before a refresh (if the value is greater than `param_theta_sample`, this parameter is ignored).
- `poisson` : `true` to set the value of the total displacement angle of the chain (resp. the refresh angle) with a Poisson law of parameter `param_theta_sample` (resp. `param_theta_refresh`).
- `epsilon_set` : The closer this value is to 0, the closer to the identity matrix are the proposed updates of the  $R$  matrix during a lift.

#### 4.4. Plaquette

- `N_shift_plaquette` : The number of shifts between two measures of the plaquette.

#### 4.5. Topology

- `topo` : `true` to measure the topological charge, `false` not to.
- `N_shift_plaquette` : The number of shifts between two measures of the topological charge.
- `N_steps_gf` : Number of Wilson Flow steps for a measure of the topological charge.
- `N_rk_steps` : Number of RK3 steps for a Wilson Flow step (the discrete flow time step is set to  $\epsilon = 0.02$ ).

#### 4.6. Save parameters

- `run_name` : Name of the save folder and save files (for example the plaquette will be stored in `run_dir/run_name/run_name_plaquette.txt`).
- `run_dir` : Path of the save folder.
- `save_each_shifts` : The number of shifts between two saves. A save is also always performed at the end of the run.

### 5. Saved files

When a save is performed (at the end of the run or every `save_each_shifts` shifts), a folder `run_name` is created at `run_dir` containing the following files/folder :

- A folder `run_name_seed` containing the state of the RNG for each MPI core of the simulation (needed for continuation of the run).
- A file `run_name` containing the whole lattice in ILDG format (needed for continuation of the run).
- A file `run_name_plaquette.txt` containing all the plaquette measurement (after thermalization).
- A file `run_name_topo.txt` containing all the Wilson Flow topology measurements (after thermalization) : the flow time  $t$ , the topological charge  $Q_{\text{clover}}(t)$  and the energy density  $E_{\text{clover}}(t)$ .

### 6. Continuation of a run

To continue a run, just execute the binary with an input file such that the `L_core`, `n_core_dims`, `run_name`, `run_dir` are unchanged. The program should retrieve in the `run_dir/run_name` folder the ILDG file and the `run_name_seed` folder necessary to continue the simulation. The other parameters of the input file can be changed as pleased.

Note that you can continue an Heatbath simulation with ECMC and vice-versa.