LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

Faculty of Physics, Summer Term 2023
Numerical Quantum Physics
Lecturer: Dr. S. Paeckel
Assistant Lecturer: Z. Xie, B. Schneider

https://www2.physik.uni-muenchen.de/lehre/vorlesungen/sose_23/nqp/

Sheet 4: Quantum Monte-Carlo

Released: 06/09/23; Submit until: 06/23/23 (**20 Points**)

On this sheet we realize an efficient, yet conceptionally simple, Monte-Carlo sampling to generate configurations of spin-$1/2$ degrees of freedom, distributed according to the Boltzmann statistics, using *cluster algorithms*. We will consider the Ising model on a square lattice in $d = 2$ dimensions with L^2 lattice sites ($L \in \mathbb{N}$, $J > 0$) and periodic boundary conditions:

$$\hat{H} = -J \sum_{\langle i,j \rangle} \hat{Z}_i \hat{Z}_j \tag{1}$$

and choose as computational basis the Pauli $\hat{Z}$-operator eigenstates $|\sigma_j\rangle$ with $\sigma_j \in \{+1, -1\} \equiv \mathcal{S}$. The goal is now to draw samples of configurations $\underline{C} \in \mathcal{S}^{L^2}$ that are distributed according to the Boltzmann distribution $p(\underline{C}) = \mathrm{e}^{-\beta E[\underline{C}]}/Z_\beta$ at an inverse temperature $\beta = 1/T$ and $E[\underline{C}]$ denotes the energy of the configuration $\underline{C}$. Once this is achieved, we can estimate observables (here functions of Pauli $\hat{Z}$-operators) by calculating expectation values with respect to a set of generated samples $\mathcal{C} \subset S^{L^2}$:

$$\langle \hat{A} \rangle \approx \bar{A} = \frac{1}{|\mathcal{C}|} \sum_{\underline{C} \in \mathcal{C}} A[\underline{C}] \; . \tag{2}$$

Problem 1 A Markov chain for cluster updates **(14 Points)**

In cluster algorithms, the partition function $Z = \mathrm{Tr}\, \mathrm{e}^{-\beta \hat{H}}$ is further expanded in a set of graphs $\mathcal{G}$, i.e., equal-time slices through world lines, that belong to the same configuration $\underline{C}$. Here, a world line is given by the evolution of an initial configuration $\underline{C}_0$

$$|\underline{C}_0\rangle \overset{p(\underline{C}_1|\underline{C}_0)}{\longrightarrow} |\underline{C}_1\rangle \overset{p(\underline{C}_2|\underline{C}_1)}{\longrightarrow} \cdots \overset{p(\underline{C}_M|\underline{C}_{M-1})}{\longrightarrow} |\underline{C}_M\rangle \; , \tag{3}$$

with $M \in \mathbb{N}$, which constitutes a Markov process. For a certain configuration and compatible graph $(\underline{C}_n, G_n)$, it is realized by performing a graph update with probability $p(\underline{C}_n, G_{n+1}|G_n)$, followed by an update of the configuration, compatible with the new graph, with probability $p(\underline{C}_{n+1}|\underline{C}_n, G_{n+1})$.

For the Ising model, this process can be obtained by noting that only neighboring spin can be flipped and there are two possible graphs associated. If under a transition $\underline{C}_n \longrightarrow \underline{C}_{n+1}$ a pair of spins is to be flipped, we denote the corresponding graph to be connected, otherwise the graph is called disconnected.

(1.a) **(3P)** Implement a lattice class which hosts a configuration $\underline{C}$ of spin-$1/2$ degrees of freedom on a two-dimensional lattice with dimensions $L \times L$. Equip the class with a method to compute the average magnetization $m = \langle \hat{Z} \rangle$. Furthermore, implement a method that initializes the lattice with a random configuration of spin.

(1.b) **(3P)** Implement an update class that acts as base class for a cluster update. It should provide an abstract method that takes as input a configuration $\underline{C}$ and returns a new configuration $\underline{C}'$. Extend your lattice class by an `update` method that takes as input argument an instance of the update class (updater) as well a number of updates N_u to be performed and which then calls the updater N_u times to generate a new lattice configuration.

(1.c) **(4P)** Derive from the basic update class to implement the Swendsen-Wang algorithm. Here, an update is defined by

- To each neighboring pair of parallel aligned spins in the lattice, assign with probability $1 - \mathrm{e}^{-2\beta J}$ a label *connected* or label it as *disconnected*.
- Find all clusters of connected spins.
- The spins in each cluster are flipped with probability $1/2$.

(1.d) **(4P)** Derive from the basic update class to implement the Wolff algorithm. Here, an update is defined by

- Choose a random spin as initial cluster.
- Label all neighboring spins that are parallel to the initial spin as *connected* with probability $1 - \mathrm{e}^{-2\beta J}$.
- Repeat the previous step for all spins that have been newly added to the cluster recursively, until the cluster is not growing any more.
- Flip all spins in the cluster.

Problem 2 Autocorrelation times **(6 Points)**

We now test the different cluster update methods by computing the autocorrelation times. These are crucial to ensure that the configurations generated by one update run can be considered to be independent, i.e., they constitute independent samples, drawn from the jont probability distribution $p(\underline{C})$. Deviations of the sample independence affect the estimation of expectation values $\bar{A}$ of observables via the autocorrelation $c_A(t)$:

$$\mathrm{Var}[\bar{A}] = \frac{\mathrm{Var}[\langle \hat{A} \rangle]}{N_{\mathcal{C}}} \left(1 + 2 \sum_{t=1}^{N_{\mathcal{C}}-1} (1 - \frac{t}{N}) \frac{c_A(t)}{c_A(0)}) \right) \tag{4}$$

with $N_{\mathcal{C}} = |\mathcal{C}|$. The autocorrelation is defined via

$$c_A(t) = \langle A_0 A_t \rangle - \langle A \rangle^2 \ , \tag{5}$$

where $\langle A_0 A_t \rangle$ denotes the expectation value w.r.t. to the 0th and tth samples of the $N_{\mathcal{C}}$ realizations.

(2.a) **(3P)** For both update methods implemented, run $N = 100$ Monte-Carlo simulations at inverse temperatures $\beta = 0.1, \ldots, 10.0$ (with a proper discretization for the β's), choosing $N_u = 100$ Markov steps and $N_{\mathcal{C}} = 50$ configurations per simulation. Start each simulation from a random configuration and use N_u initial updates to erase the memory of the initial state (burn-in). In every simulation after each cluster update, measure the cluster's magnetizations m and store the sequence of $N_{\mathcal{C}}$ measurement results.

(2.a) **(3P)** Compute the autocorrelation $c_m(t)$ for both update schemes. Estimate the integrated autocorrelation time $\tau_m = \sum_{t=1}^{N_{\mathcal{C}}-1} (1 - \frac{t}{N}) \frac{c_m(t)}{c_m(0)})$ and compare it to the sample variance of the magnetization.