Exercise 2: Swendsen-Wang algorithm for the 2D Ising model

The goal of this exercise is to implement the Swendsen-Wang algorithm for the 2D Ising model. Recall that the update of the Swendsen-Wang algorithm works as follows:

- Assign to each bond b a weight $w_b \in \{0, 1\}$, where 1 means "connected" and 0 means "disconnected". If the spins connected by the bond $b = (n, m)$ are antiparallel, always choose $w_b = 0$, if the spins are parallel, choose $w_b = 0$ with probability $e^{-2\beta J}$, where $\beta = \frac{1}{k_B T}$. In terms of conditional probabilities:

$$\begin{aligned} p(w_b = 0|\sigma_n \neq \sigma_m) &= 1, & p(w_b = 0|\sigma_n = \sigma_m) &= e^{-2\beta J}, \\ p(w_b = 1|\sigma_n \neq \sigma_m) &= 0, & p(w_b = 1|\sigma_n = \sigma_m) &= 1. - e^{-2\beta J}. \end{aligned}$$

- Interpreting these weights as edges of a graph (where 0 means no edge), find "clusters" of spins, i.e., connected components of the graph.
- Flip each cluster (= connected component) with probability $p = 0.5$.

After the update, do a measurement (if the system is already thermalized) and continue with the next update.

a) Since we need to identify clusters, it is better to label the lattice sites (x, y) by a single integer number $n = n(x, y) = x \cdot L_y + y = 0, 1, \cdots, N - 1$ where $N = L_x \cdot L_y$. The inverse mapping is given by $x(n) = n//L_y$ (integer division, rounding down) and $y(n) = n \mod L_y$. Create an array that contains each bond (n, m) of the square lattice with periodic boundary conditions exactly once. The bond array should have shape $(2N, 2)$.

b) Initialize a 1D array for the spin states σ_n (having random entries ± 1).

c) Write a function that determines the weights w_b for each of the bonds.

d) To determine the connected components, you can use the function `scipy.sparse.csgraph.connected_components`. Look up the documentation of the function online. The graph is represented by a sparse matrix of the type `scipy.sparse.csr_matrix`. You can use the initialization of the form `csr_matrix((weights, (bonds[:, 0], bonds[:, 1])), size=(N, N))`. Don't forget to add the transposed to obtain a symmetric graph. Using the result of `connected_components`, flip each of the determined clusters with probability $p = 0.5$. Collect the code in a function performing one update with the Swendsen-Wang algorithm.

e) Write functions to measure the energy and magnetization.

f) Write a function to run the whole Monte Carlo simulation at given temperature, returning arrays with all measured values E and M. Plot the mean values for different temperatures and make sure you get (roughly) the same results as last week. (Begin the comparison with small systems!)

g) Optionally: Now is a good time to optimize the code with `numba.jit`.

h) Measure and plot the autocorrelation of the energy

$$C_E(\delta) = \frac{\langle E_{t+\delta} E_t \rangle_t - \langle E_t \rangle_t^2}{\langle (E_t)^2 \rangle_t - \langle E_t \rangle_t^2}$$

versus δ (and similarly for the magnetization) for T right at, above, and below the critical temperature T_c.

i) Recreate the the autocorrelation plots for an update function which proposes to flip $L_x \cdot L_y$ random spins with the Metropolis algorithm.