Exercise 3: The 2D Ising model

Goals

The Ising model is a well-known way of describing ferromagnetism using simplified spin-spin interactions. We will construct a 2D Ising model and use it to measure the thermodynamic and magnetic properties of the system as a function of time and as a function of the temperature and magnetic field.

Physics

The 2D version of the Ising model considers an w by w square lattice, resulting in $N = w^2$ spin sites. It is conventional to use *periodic boundary conditions*, where the lattice "wraps round" from the left edge to the right edge and from the top edge to the bottom edge; topologically the system is then the same as the surface of a toroid. With these boundary conditions every spin has 4 neighbours. The energy of the system is then given by

$$E = -J \sum_i \sum_j s_i s_j - \mu H \sum_{i=1}^{N} s_i$$

where we sum over all spins s_i and their neighbours s_j with exchange energy J, magnetic moment μ, and externally applied magnetic field H.

The most interesting variable of the system is the relative magnetisation of the complete lattice

$$M = \frac{1}{N} \sum_{i}^{N} s_i$$

or the expected total magnetisation

$$\mathbb{E}[M(\mathbf{s})] = \sum_{s \in \{-1,1\}^N} M(s) p_{\mathbf{s}}(s)$$

with $p_{\mathbf{s}}(s)$ the probability mass function. The magnetisation shows a very strong dependence on the temperature as well as the externally applied magnetic field H. The magnetisation can also show a high variance.

The analytical approach to calculate these expected values would be to use the probability mass function

$$p_{\mathbf{s}}(s) = \frac{1}{Z} e^{-\beta E}$$

with partition function

$$Z = \sum_{s \in \{-1,1\}^N} e^{-\beta E}$$

and inverse temperature $\beta = \frac{1}{k_b T}$. However, computing this type of partition function quickly becomes arduous or simply infeasible for a lattice with any interesting number of spins, especially if the variance is also high. We thus typically use Monte Carlo methods to investigate such systems and estimate the magnetisation M.

The Metropolis-Hastings algorithm The basic algorithm is as follows: we construct a w by w grid of spins, starting from some initial set of spin orientations, perhaps a random or uniform set, the model is evolved in time by a Monte-Carlo technique: pick a spin, and calculate the energy ΔE required to flip it. If $\Delta E < 0$, flip the spin; if $\exp(-\Delta E/(k_B T)) > p$, where p is a random number drawn from a uniform distribution in the range $[0, 1]$, then flip the spin; else do nothing.

We now repeat this step for many lattice sites. One can either step systematically through the lattice doing N possible flips, or pick N random points. You can think of these N operations together as a single *time step*. We repeat many steps to evolve the spin system. We wait for the system to reach equilibrium and then sample relevant quantities and average them over a large number of time steps.

Tasks

Core Task 1: No spin coupling Write a program to construct a lattice and evolve it with time according to the Metropolis-Hastings algorithm. It is probably simplest to parameterise your functions in terms of the dimensionless quantities βJ and $\beta \mu H$.

Set $\beta J = 0$ so there is no coupling. In such a case, only thermal motion and the external magnetic field affect the flipping of the spins. Do a few simulations with $-3 < \beta \mu H < 3$ and observe what happens by plotting the lattice configuration after 1, 10, 100, 1000, 10,000 and (if you have time) 100,000 "time steps" (you can display the lattice configuration using the `matplotlib.pyplot.matshow()` function). Try both starting with a random lattice and a lattice with all spins in one direction.

Compute the relative magnetisation M of the lattice and plot it as a function of time for at least 10,000 steps. Qualitatively explain the system behaviour.

In the case of no coupling, an analytical expression for the mean magnetisation $\langle M \rangle$ exists. The expression is:

$$\langle M \rangle = \tanh(\beta \mu H)$$

Test whether the Monte Carlo method correctly reproduces the above analytical formula.

Core Task 2: only spin coupling Now let us investigate coupling without an external magnetic field ($H = 0$). Try the same as previously, using different starting conditions, but introduce some ferromagnetic coupling ($\beta J = 0.2$) or anti-ferromagnetic coupling ($\beta J = -0.2$). Qualitatively explain the system behaviour.

For this case, no field and only coupling, Lars Onsanger discovered an analytical solution for the case of an infinite lattice:

$$\langle M \rangle = \begin{cases} 0 & \text{if } T \geq T_c \\ \pm \left(1 - \sinh\left(2\beta J\right)^{-4}\right)^{\frac{1}{8}} & \text{if } T < T_c \end{cases}$$

where T_c is the critical temperature given by

$$T_c = \frac{2J}{k_b \ln(1+\sqrt{2})}.$$

Test whether the Monte Carlo method correctly reproduces the above analytical formula. Comment on any unusual behaviour of the system you notice around the critical point.

Hint: The system will equilibrate much faster if you start with an initial state that is close to the expected equilibrium state. This can be achieved, for example, by using the final lattice state from a simulation at one temperature as the starting point of a new simulation at a nearby temperature.

Supplementary task: Investigate the heat capacity near the phase transition. The fluctuation-dissipation theorem states that the heat capacity of the system is given by

$$C = \frac{\sigma_E}{k_B T^2}$$

with σ_E being the standard deviation of fluctuations in the system energy E. Plot a graph of the heat capacity of the system as a function of temperature for $H = 0$, evaluating what happens close to the critical temperature T_c for a number of different lattice sizes.

Hints

Results get more representative of a real ferromagnetic system as we go to larger and larger lattice sizes w, but larger lattices take more computing time, so make sure you write functions which can generate and evolve an arbitrary-sized (but square) lattice, and then experiment with different-sized lattices to see how large a lattice is practical.

At the same time, to get accurate values for mean quantities such as the magnetisation requires that (a) we wait till the system has reached equilibrium before starting averaging, and (b) that we average over many independent microstates after we reach equilbrium. Both these requirements mean that

the increasing the speed at which we can evolve the system is critical so that we can generate many thousands of time steps in a reasonable time.

Since this is Python, the majority of the speed gains we can achieve are by vectorisation. Vectorising the Metropolis-Hastings algorithm takes some thought, but can be done. One way to do this is to notice that, if we imagine mapping the cells of the lattice on to a chess board pattern, then all the black squares can be evolved at the same time, because evolving one black square does not affect the nearest-neighbours of any of the other black squares. The same is true of the white squares, so in principle we can do one "time step" by doing all the black squares, then all the white squares. Accessing all the black squares or all the white squares simultaneously can be done by generating an array of boolean values to "mask" certain lattice sites. Read up on numpy boolean masking to understand how this works!

The `numpy.roll()` function is also useful for vectorising as it allows us to get one of the nearest-neighbour spins for all cells in the lattice in a single function call.

More information on Metropolis-Hastings and other algorithms for solving the Ising model can be found at:

https://hef.ru.nl/~tbudd/mct/lectures/markov_chain_monte_carlo.html

https://hef.ru.nl/~tbudd/mct/lectures/cluster_algorithms.html

This exercise is based on an exercise developed by one of the demonstrators on this course, Danny van der Haven.