The Ising Model through Montecarlo method

Brief theoric overview of the ferromagnetism and Ising Model

An atom in a lattice is of course supposed to make bonds. How will the spins be arranged after? The resulting molecular orbital will thus be bonding or anitbonding; as the name suggests the bonding(where spins allign antisimmetrically) state is the most energtically favoured to create bonds. This happens because of the simmetricity of the radial-total wave function(the total wave function must remain antisimmetric). So what happens in Fe?
We know Fe to form bonds with its further shell 4s; the electrons in this shell will combine in order to be arranged in a singlet state since it's the most energetically convenient to form bonds; the orbital 3d also overgoes an exchange interaction but in this case things go differently because the exchange integral(a physical quantity underlying the orbital's overlapping) is positive hence making the triplet state the most energetically favoured. That causes spins to allign parallelly. The single-electron spin exchange hamiltonian(being J the exchange interaction and S_i, S_j the spins) reads like:

$$H_i^{exchange} = -2 \sum_{i \neq j} J^{Exchange} S_i S_j \tag{1}$$

A different approach to understand magnetic behavoiur of solids is to consider particulat microscopit models for magnetic interaction. can be simplified by simply considering the sum as performed throughout the nearest neighbours. In the Ising models spins are only allowed to point up or down, i.e. we only consider the x component of the spin. The hamiltonian will hence read like :

$$H_{Ising} = - \sum_{nearestneighbours} J S_i S_j \tag{2}$$

The code I wrote focused on a 2D lattice made by N*N sites which spins are placed on.To maximize the interaction of the spins at the edges of the lattice they are made to interact with the spins at the geometric opposite edges of the lattice.
This is referred to as periodic boundary condition (pbc) and can be visualized better if we consider the 2d lattice being folded into a 3d torus with spins being on the surface of such a topological structure.
The system will undergo a phase-transition from magnetic-reached order to magnetic disorder above a certain crytical temperature T_c(we suppose to operate in absence of external magnetic field). Once reached the order, the system will tend to remain in the same state since the energetic cost to break the order phase will be too high.
As order parameter for the phase transition we 're going to use the absolute magnetization, defined as:

$$M = \frac{<m>}{N*N} \tag{3}$$

Metropolis algorythm

The algorithm will proceed as follows:

- **Configuration**: we are going to initialize the NxN lattice with a random configuration of spins, hypotizing in can only assume the values [-0.5,+0.5] related to the different orientation of the spin along the third dimension.

- **propose a move**: we are going to randomly choose a site of the lattice and we are going to compute the change in energy dE we would have if we flipped the sping. Considered the Ising hamiltonian, this quantity will be $dE = 2 * S_{site} * \sum_{nn} S$

- **Accept or refuse the move** : The move we proposed will be accepted or refused according to the Metropolis algorythm. If dE¡0 the move will be always accepted since it reduces the energy of the system. if dE¿0 we can accept the move only if a generic random number between 0 and 1 is lesser than e^{-dE/k_BT} being k the Boltzman constant which will be set to 1.

Of course we are going to loop over a consistent number of steps and continously repeat the last three points.

How the repository works

Metropolis algorythm applied to electoral laws

A good electoral law should be a decent balancement of both representivity and governability. The first principle can lead to proportional laws, the second to majoritary law. So in the last years electoral laws mixing these two features were experimented all over european democracies.
Let's take into account the current italian electoral law and let's focus only on the election of "La Camera dei Deputati". There are 630 deputates in the "small chamber". The 61% is elected in a proportional way(party x gets N% votes, it gets 0.N*0.61*seats). The remaining 37% is elected thorough the collegia mechanism. This means the italian territory is split in 212 electoral collegia. In each collegium different electoral coalitions(or single parties) fight to "win that collegium". In fact, if X coalition/party reaches the highest percentage in the land which represents the collegium it will take that seat in the Parliament.
The question rising up is the following: is it possible to foresee the assignment of the seats in the Parliament, only knowing the national results? The answer, as this program seems to prove, is yes.
The most obvious answer would be to assign the number of collegia according to the national results (X party gets N% votes, it will have N*0.37 seats in the Parliamento). This couldn't be further from the truth because if a party gets a score of 40% it is almost sure that it's going to win more than the 40% of collegia.
Immagine we have three main coalitions (A = 40%,B = 30%,C =20%) and let's consider a single collegium. If we consider the results as probability weight, we can consider to moltiply every score to a random number between 0 and 1. The seat will be obtained by the highest score vehiculated by a random number.
If we repeat this procedure over all the collegia, eventually we will be able to map all the national territory and to reproduce with decent accuracy the national results.
Of course there are many flaws to this algorithm which is totally unable to foresee very localized electoral exploits of some regional parties. So if a party gets the 1% on national scale and in a specific region with 3 collegia it has the 50% it will get 3 seats and no way to foresee that. Actually if we run very long simulations (more than a million times) a

single anomalous result like that can be reproduced.
The algorithm might also be used to highlight some shortcoming of the specific electoral law we're dealing with. It might happen(it sounds absurd) that two parties obtaining almost the same results on national scale will have a completely different assignment of seats. Montecarlo could see that. In fact it is provided an histogram in the graphic part in which the distribution of different electoral simulation for each party are confronted. There is a slightly possibility that a party which got a lesser result could benefit a larger assignment of seats(overlapping between the tails'distributions).
The appropriate number of steps for each simulation can be determined by knowing how many steps need to be performed to compute pi by exploiting Montecarlo methods.

How the repository works

in the file ***Electoral_Montecarlo.py***, all the most important functions are defined.