Ising Model

Eoin O'Connor

January 15, 2018

Abstract

Various different lattices were analysed in this project using the Ising model. These include the square lattice, bcc, fcc and triangular lattices.The critical phase transition temperature was found for each of the lattices, for the square lattice this was $2.25 \pm 0.1 J/k_B$. The square lattice and triangular lattice were both animated using blitting to increase efficiency.

Introduction and Theory

The Ising model is used to model ferromagnetism in statistical mechanics. The model uses a lattice in which each point is assigned a spin, either up or down. Each point in the lattice can interact with it's nearest neighbours. The Hamiltonian of the system is given by

$$H(\sigma) = -\sum_{\langle i\ j \rangle} J_{ij}\sigma_i\sigma_j - \mu \sum_j h_j \sigma_j \tag{1}$$

Where σ_k corresponds to the spin of the kth point in the lattice. It has a value of +1 for an up spin and -1 for a down spin, J_{ij} is the interaction strength between the ith and jth spin. μ is the magnetic moment and h_j is the interaction of the external magnetic field with the jth point.

The lattice is said to be ferromagnetic if all of the spins point in the same direction. The lattice is said to be antiferromagnetic when all the nearest neighbours of a given lattice point in the opposite direction to itself. In this assignment different lattices, different values for J_{ij} and h_j and different values of temperature are examined to find what conditions lead to ferromagnetism and antiferromagnetism.

The probability that the system is in a state ν is given by the boltzmann distribution

$$p_\nu = \frac{e^{-\beta E_\nu}}{Z} \tag{2}$$

Where $\beta = \frac{1}{k_B T}$ and Z is the partition function given by $Z = \sum_\nu e^{-\beta E_\nu}$. The probability that the system changes from a state y to a state x is given by

$$\frac{P_x}{P_y} = P_{flip} = e^{-\beta \delta E}$$

Where $\delta E = E_x - E_y = H_x - H_y$

Experimental Method

The majority of the experimental method is explained in the comments of the code. Feel free to change the initial conditions in any of the programs. All programming was done on a vertual machine running Ubuntu. All analysis and graphs were done on windows because the programs run quicker and the graphs look nicer.

The first file that was created was *isingmatrix.py*. This program uses fixed boundary conditions and a simple algorithm that iterates across the entire matrix then switches all the necessary spins. The next program that was coded was *periodicbound.py*. *isingmatrix.py* was taken and the fixed boundary conditions were replaced with periodic boundary conditions. The ability to apply an external magnetic field was also added at this stage.

Next the simple algorithm was replaced with the metropolis algorithm in *metropolis.py*. This algorithm is explained in Figure 1. The algorithm is run a large number of times in order to bring the lattice to equilibrium. The next program that was created was *blit_attempt.py*. The intention of this program was to animate the evolution of the lattice as the algorithm works on it. Unfortunately the save feature of the animations wasn't saving properly but the animation can be viewed by running the code.

metropolis.py was extended on to test multiple temperatures and analyse the resulting lattices for properties including magnetization and energy per site in *analysis.py*. This program is commented extensively as it is the basis for most of the subsequent programs. *onedim.py*, *threedim.py*, *bcc.py* and $fcc.py$ are all slight variations of *analysis.py* with the nearest neighbours changed.

In order to further analyse the triangular lattice an animation was made of it using circular patches. Unfortunately the circular patches are much more graphically intensive and can only be done for smaller latices.

Bash scripting was used to increase ease of use when coding the project. *quickgit.sh* allowed a file or folder to be committed to gitlab with minimal effort. *newcode.sh* creates a new file, gives it executable permission and opens in gedit all in one go.

Results and Analysis

Due to the fixed boundary conditions *isingmatrix.py* was not very useful or efficient as reaching equilibrium. Even with the periodic boundary conditions added *periodicbound.py* was still not efficient at reaching equilibrium, failing to bring a 20x20 matrix to equilibrium even after 2 minutes of computation.

In contrast *metropolis.py* is consistently able to bring a 20x20 matrix to equilibrium in under 0.2 seconds. It was a clear choice to continue the rest of the project with this algorithm. Figure 2 shows that the metropolis algorithm brought the 70x70 matrix to equilibrium in less than 3 million iterations. Obviously it is possible for the program to take longer or shorter than this based on the initial configuration and the random walk. The program took less than 14 seconds to run those 4 million iterations.

The animation shows that initially the lattice is random then begins to arrange itself into long narrow domains as seen in Figure 3. This process is relatively quick compared to the overall equilibrium process. Eventually the domains begin to join up and two large domains are seen. If one domain is a lot larger than the other then it will surround the smaller one and consume it. If this happens the program takes relatively few steps to reach equilibrium. If the domains are relatively equal in size then they will orientate themselves so that the boundary between them is either vertical or horizontal. Then there is a sort of tug of war between the two domains until one eventually consumes the other. This process can take a very long time and the metropolis algorithm is maybe not the most efficient way of reaching equilibrium from this state.

The graphs in Figure 4 show that there is a significant fall off in magnetization of the lattice at a temperature of around 2.25 ± 0.1. This is also the point where the energy is increasing fastest, where the specific heat capacity peaks and where the magnetic susceptibility peaks. Therefore it is clear that the critical temperature of the 2d square lattice is 2.25 ± 0.1. The ground state energy of the lattice is -2. The only limit on the accuracy of the results is the computation power and time allocated. With enough computation time the drop in the magnetization would likely be sharp resulting in an infinite peak of the magnetic susceptibility at the critical temperature.

The results from Figures 5, 6, 7, 8 and 9 are shown in the table below:

Lattice	Critical temperature	Ground state energy	Nearest neighbours
1d	0.4	-1	2
2d Square lattice	2.25±0.1	-2	4
Simple cubic	4.3±0.1	-3	6
Body centered cubic	5.5±0.1	-4	8
Face centered cubic	9.5±0.1	-6	12
2d Triangular lattice	3.4±0.1	-3	6

There is clearly a general trend that the ground state energy is equal to to the number of nearest neighbours divided by -2

The 1d lattice shows a critical temperature of 0.4 but also when smaller lattice sizes were considered this value went up, it is likely that with larger lattice sizes this value will further approach 0 as is predicted theoretically.

For the simple cubic, body centered cubic and face centered cubic lattice an 8x8x8 lattice was used to reduce computation time. The critical temperature certainly increases with the amount of nearest neighbours but there is no clear equations that can be seen for the relationship. The computation time for the 3d lattices, especially the fcc lattice scales very quickly with larger lattice sizes. The graph for the fcc lattice took 100 minutes to produce. The Ising model for dimensions $n \geq 2$ has been proven to be np-complete. This means that the time it takes to get accurate results for the model scales exponentially with time.

The triangular lattice shows once again that the number of nearest neighbours has a correlation to the critical temperature although it is still significantly less than the simple cubic which has the same number of nearest neighbours. The domains in the triangular lattice certainly tend to arrange themselves in more of a triangular or hexagonal shape when compared to the square lattice. The boundary between the two large domains is also a lot less linear in the triangular lattice.

Conclusions

The metropolis algorithm is clearly the superior of the two algorithms used being at least 1000 times quicker to reach equilibrium. The critical temperature for the square lattice Ising model was found to be 2.25 ± 0.1 which agrees with the theoretical result of 2.269 within error. A clear relationship was found between the ground state energy of a lattice and the number of nearest neighbours it was. That is the number of nearest neighbours is equal to -2 times the ground state energy. The critical temperature also increases with the number of nearest neighbours but no clear relationship can be seen. The 3d lattices took longer than expected to compute, given a longer time it would be easy and interesting to analyse the lattices with a negative J value or with a non-zero h value.

References

http://www.tcd.ie//Physics/people/Tom.Archer/teaching/computational_methods/lectures/lecture9.pdf
http://users-phys.au.dk/fogedby/statphysII/no-PT-in-1D.pdf
http://abel.math.umu.se/ klasm/Uppsatser/cubes.pdf

Appendix

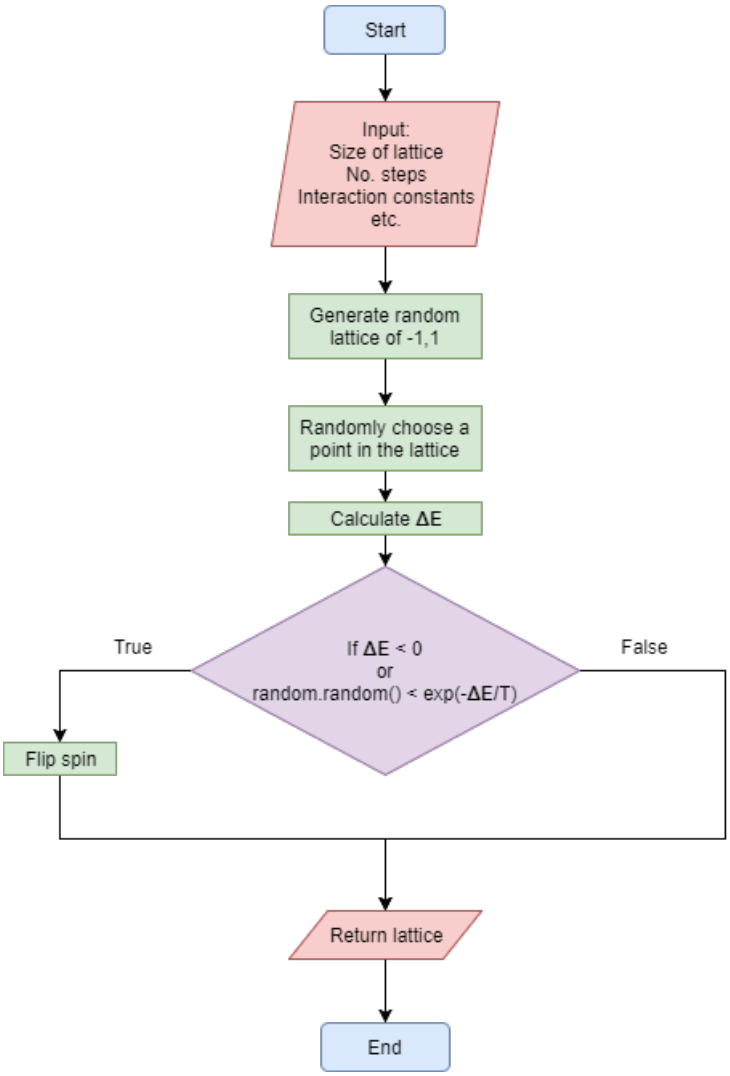

Figure 1: Metropolis algorithm

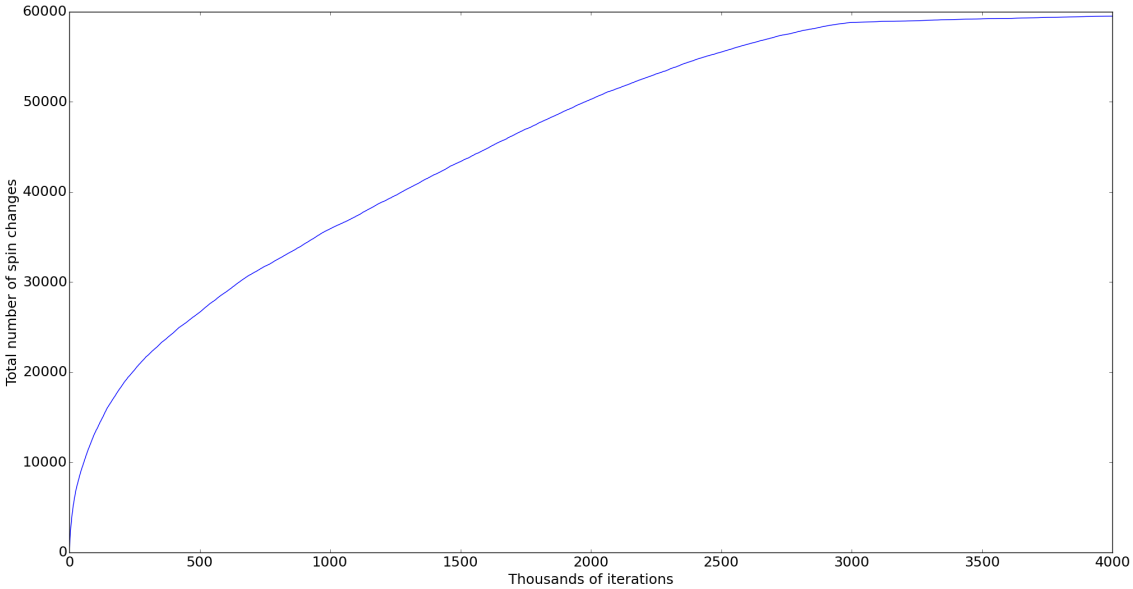

Figure 2: Spin changes vs steps for a 70x70 matrix

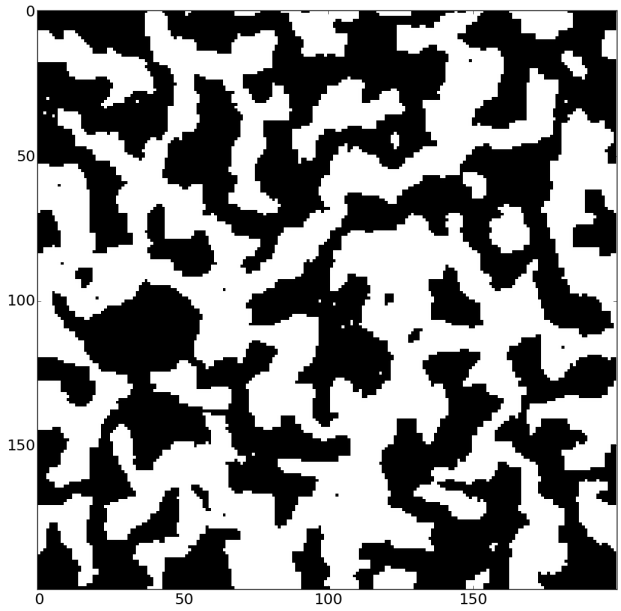

Figure 3: Long narrow domains

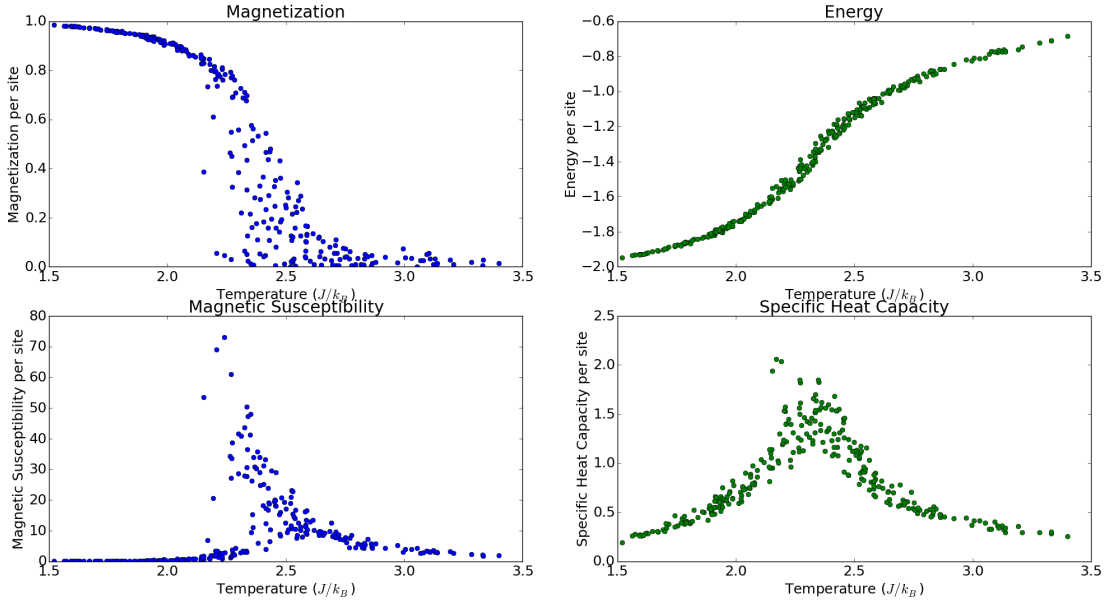

Figure 4: Analysis of the 2d square lattice

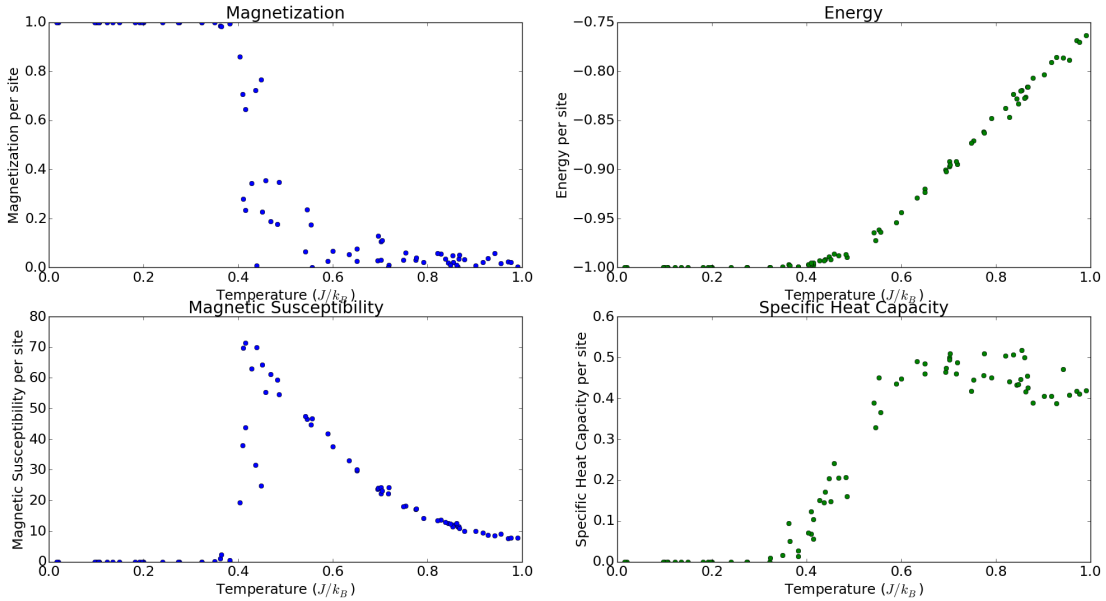

Figure 5: Analysis of the 1d lattice

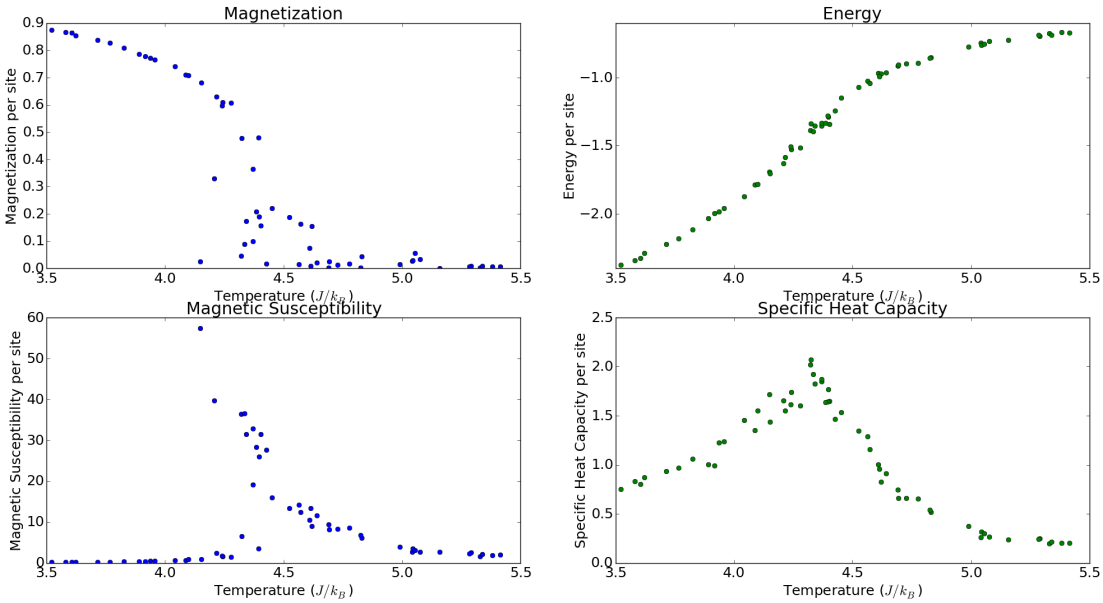

Figure 6: Analysis of the simple cubic lattice

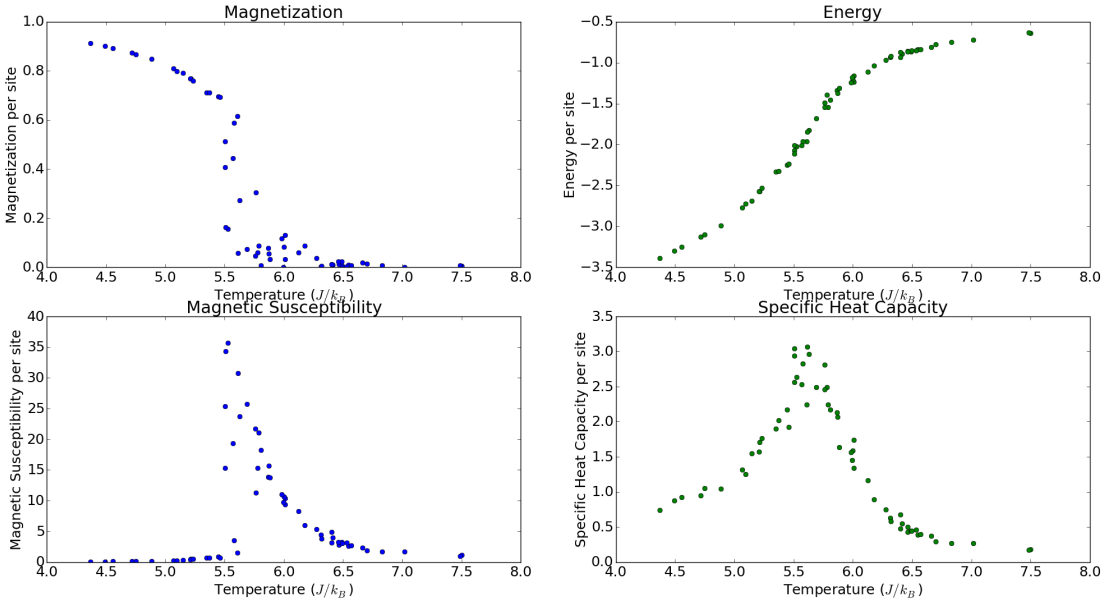

Figure 7: Analysis of the body centered cubic lattice

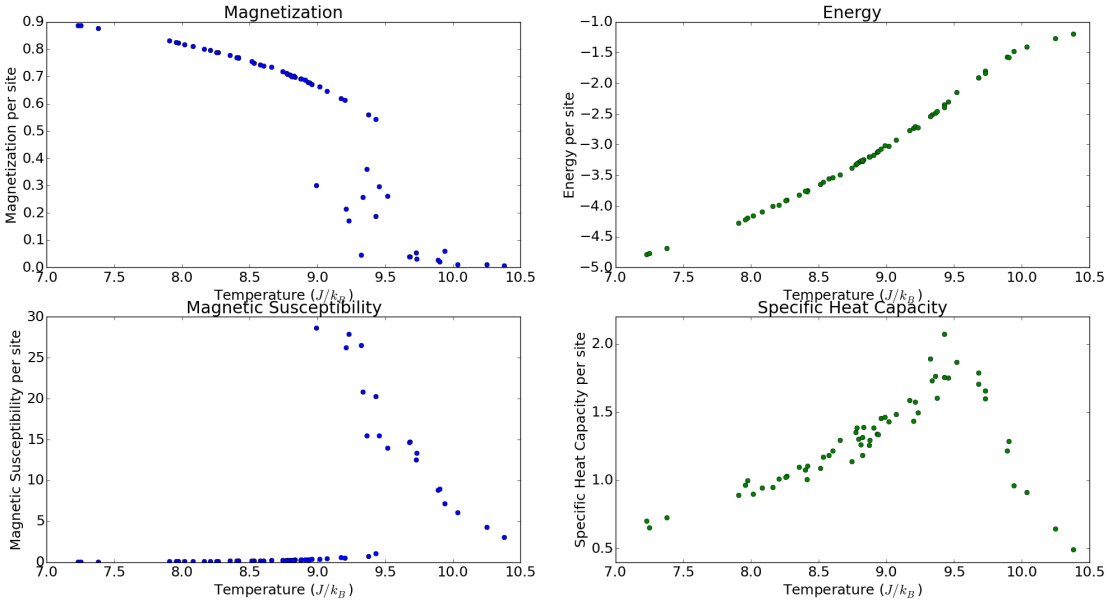

Figure 8: Analysis of the face centered cubic lattice

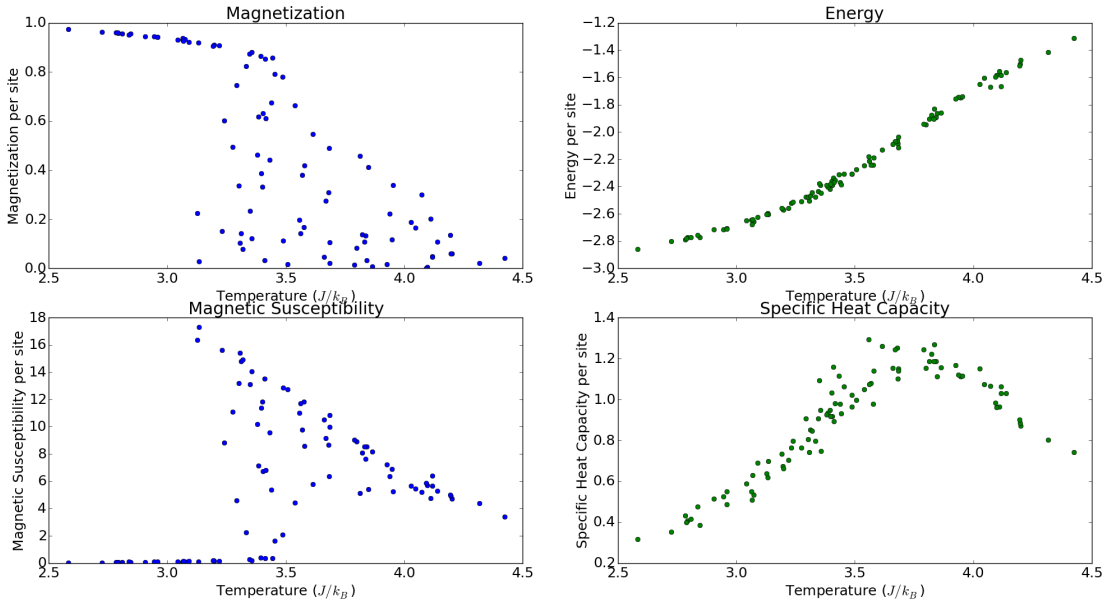

Figure 9: Analysis of the triangular lattice