The Ising model

Julia Rowe, Caleb Helbling, & Andrew DeBenedictis

3/30/2015

In this paper, we present two approaches used for solving the Ising Model in order to explore the dependence of the phase transition temperature of a ferromagnetic material on the number of neighbors of each lattice site. We implement the Metropolis algorithm and the Wolff method to relax a set of initially random spins assigned to sites in a lattice. We first consider a 1D square lattice in which each lattice site has 2 nearest neighbors (NN), then a 2D square lattice (4 NN), a 3D square lattice (6 NN), and a 2D triangular lattice (6 NN). Algorithms are executed in Python and visualized in Mathematica.

Not shown in this document are the example simulations that show the relaxation of the spins in a lattice over time. These (extremely cool) visualizations can be found in the supporting material Mathematica notebook “Ising visual.nb”.

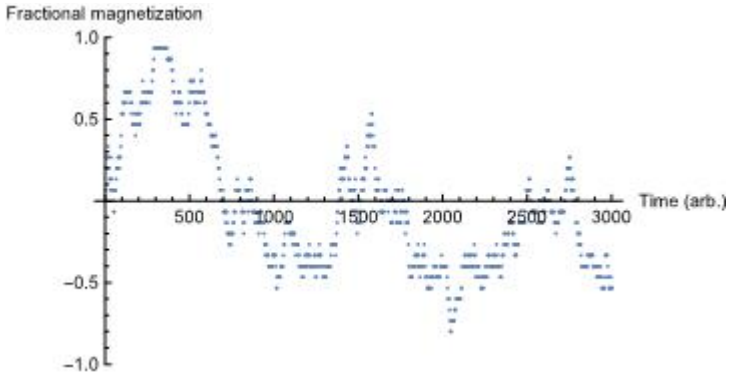

Figure 1: Relaxation time for a sample simulation of a 1D square lattice with kT=0.5.

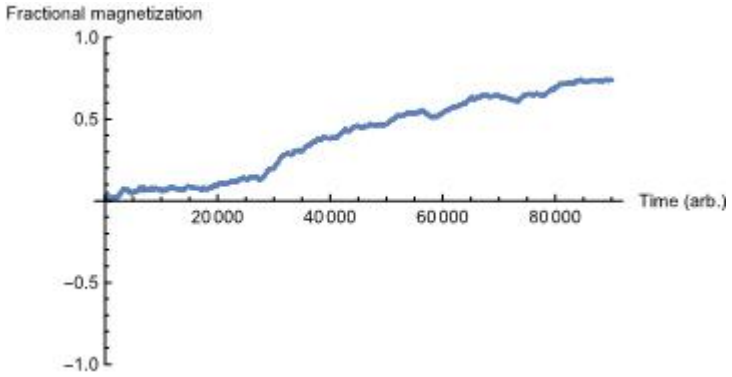

Figure 2: Relaxation time for a sample simulation of a 2D square lattice with kT=0.5.

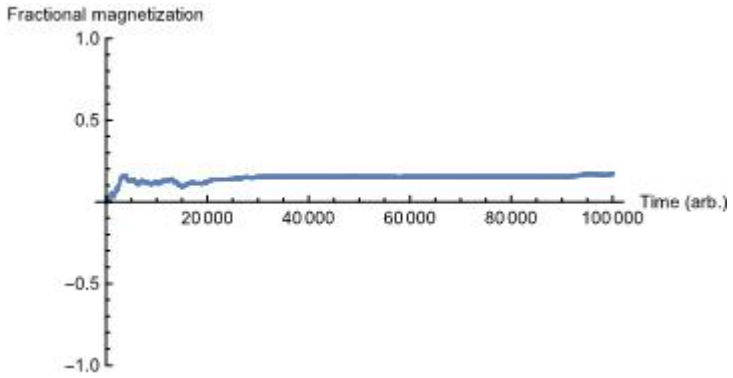

Figure 3: Relaxation time for a sample simulation of a 3D square lattice with kT=0.5.

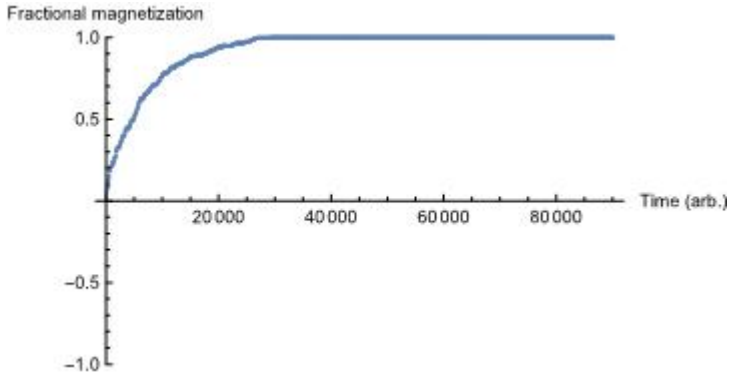

Figure 4: Relaxation time for a sample simulation of a 2D triangular lattice with kT=0.5.

Figures 1 through 4 show the evolution of a lattice's magnetization as a function of iteration number. For a lattice below a certain critical temperature, we see that, given enough iterations of the algorithm, the magnetization reaches a state of equilibrium at either $M = 0$ (for an antiferromagnetic material,$J > 0$) or $M = N$ (for a ferromagnetic material, ($J < 0$)) where N is the total number of lattice sites. This equilibrium is achieved more rapidly in higher dimensions and with a triangular lattice, indicating that the speed at which equilibrium is reached is directly proportional to the number of neighbors.

By running several cases and looking at the final magnetization after a sufficiently large number of iterations, we are able to view the equilibrium magnetization as a function of temperature for lattices of different dimensions. Because the probability of flipping a spin goes as

$$e^{2NN/kT}$$

for a lattice site surrounded by identical spins (identical to one another, not necessarily to the spin of the site in question), we expect the equilibrium temperature to depend strongly on the number of NN of each site in a given lattice. In fact, we expect the relationship

$$3kT_{c,1} \simeq \frac{3}{2} kT_{c,2} \simeq kT_{c,3}$$

to approximately hold for a square lattice, simply based on the number of NN of each arrangement. Here $kT_{c,i}$ is the critical temperature for a lattice of i dimensions. However, our results do not support this hypothesis. There must be a bit more going on which we did not have adequate time to fully explore.

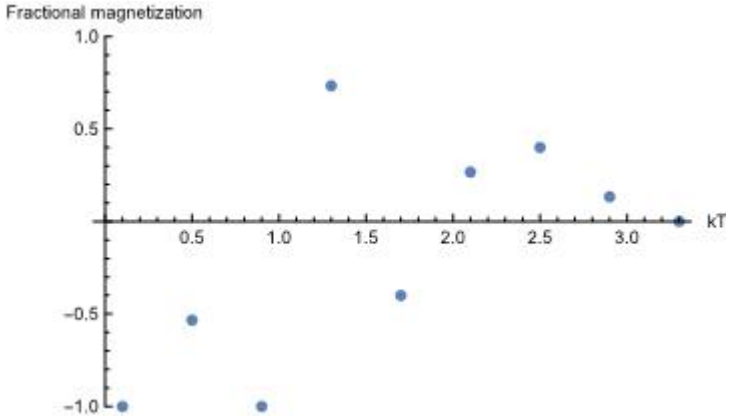

Figure 5: Magnetization as a function of temperature for a 1D square lattice.

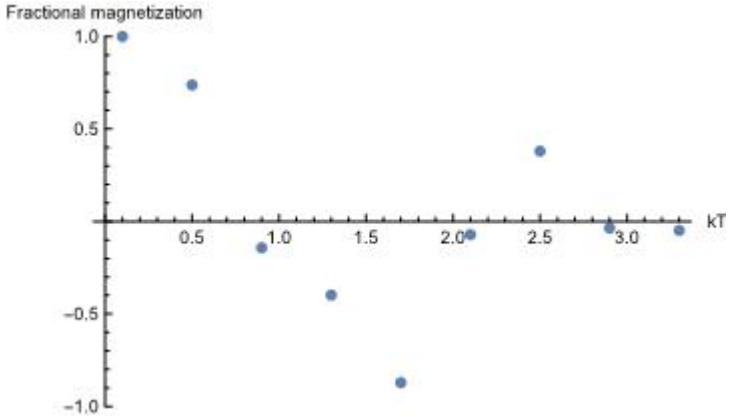

Figure 6: Magnetization as a function of temperature for a 2D square lattice.

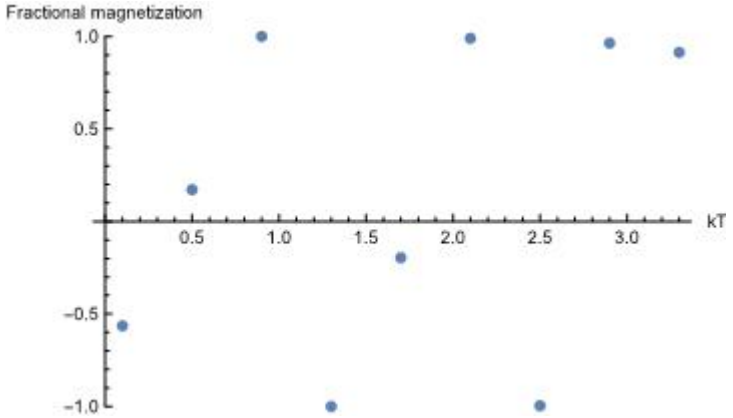

Figure 7: Magnetization as a function of temperature for a 3D square lattice.

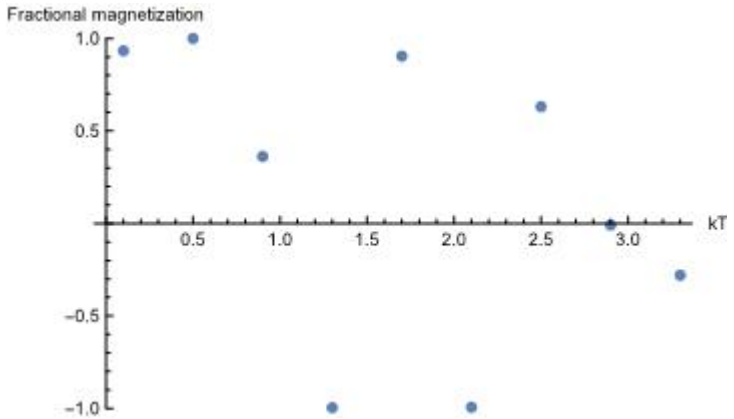

Figure 8: Magnetization as a function of temperature for a 2D triangular lattice.

Figures 5 through 8 show the magnetization as a function of kT for each lattice explored. We note that the critical temperature does increase with the number of NN as expected.

Appendix: List of Ising model algorithms

Metropolis method
Sources:

- http://en.wikipedia.org/wiki/Ising_model
- http://cs.adelaide.edu.au/~paulc/teaching/montecarlo/node29.html

Method:

1. Pick a site based on some distribution function (there are a few different ways to do this) and calculate its contribution to the energy
2. Flip the spin and calculate the new energy
3. Keep the new configuration with probability MIN[$e^{-(H_2-H_1)/kT}$,1]
4. Repeat

Pros:

- algorithm is fast for sparsely connected grids (because only nearest neighbor interactions influence choice)

Cons:

- possible autocorrelation issues
- “burn-in” period is often needed

Wolff method
Sources:

Method: Similar to Swendsen-Wang. The difference between these two algorithms is that in the Swendsen-Wang algorithm every connected component of the generated cluster could change the spin, while in Wolff ’s algorithm only one connected component will be changed surely.

Pros:

Cons:

Swendsen-Wang
Sources:

- http://www.inference.phy.cam.ac.uk/mackay/itila/swendsen.pdf
- https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=3&cad=rja&uact=8&ved=0CDIQFjAC&url=http%3A%2F%2Fopus4.kobv.de%2Fopus4-zib%2Ffiles%2F4218%2Fwende_steinke.pdf&ei=oVH_VIzvLM2syASjyoHACA&usg=AFQjCNHRKL_B4tStR2WFg-MzP_g2_2FFJw&sig2=2QtA6PqOQZn9KuMLOXmthQ

Method:

1. consider a system in which we have both the spin variables and bond variables/ the bond variables are between each pair of connected sites and take values 1 (open) or 2 (closed)
2. update bonds
    a. bonds are determined by a probability function that includes T (see 2nd link)
3. flip entire clusters to same spin (equal prob for either direction, i think)
4. repeat from 2

Pros:

- possible parallelization candidate
- cluster methods do not slow down near criticality

Cons:

- slightly harder to code

Gibbs sampling

Sources:

- http://people.csail.mit.edu/rameshvs/content/ising-gibbs.pdf

Method: Supposedly similar to the Metropolis method, but I don't really understand it. You fix all except one randomly chosen variable. Samples that variable. Then repeats for more random variables. Uses conditional distributions (but I'm not sure how). I think you look at the probability of a flip given all of the other points. Then I guess flip if greater than .5

Pros: similar to metropolis - could reuse some code

Cons: seems very similar to metropolis. also the lack of understanding.

Heat-bath

Sources:

- https://statmechalgcomp.wikispaces.com/Heat-bath+algorithm+for+the+Ising+model

Method:

Pros:

Cons: also a single flip algorithm

Herrmann method (unofficial name)

Sources:

- http://download.springer.com/static/pdf/850/art%253A10.1007%252FBF01033083.pdf?auth66=1426012950_5f2c907031467f2f9398b3c29d023820&ext=.pdf

Method:

1. divide the lattice into two sublattices (checkerboard style) A & B
2. assign a random number of spins in one sublattice to be up. set all other spins to down
3. for sublattice A, determine which spins will flip at time t+1
    a. use same spin flipping selection as Metropolis algorithm
    b. (other source says flip all spins where sum of nearest neighbor spins is 0, but i don't see the motivation for this)
4. repeat for sublattice B
5. repeat steps 3 and 4

Pros:

- deals with entire sublattice at a time, so should be quick

Cons:

- not sure how this works for a non-square lattice

Multiple-range random walk (unofficial name)

Sources;

- http://journals.aps.org/prl/pdf/10.1103/PhysRevLett.86.2050

Method:

1. start with $g(E) = 1$, and modification factor $f = e^1$
2. select random site and flip spin with probability $p(E_1 -> E_2) = Min[g(E_1)/g(E_2),1]$
3. set $g(E) = fg(E)$for new energy state
4. repeat until H(E) is "flat" (defined as all H(E) >= 0.8<H(E)>)
5. then set $f = (f)^{1/2}$and $H(E) = 0$and return to step 2
6. continue until f is within some specified distance of 1 (very close)

Pros:

- supposedly very fast
- has tunable parameter to set tolerance of results (converges accurate to ln(f)

Cons:

- detailed balance not satisfied until later stages of simulation
- i don't yet understand how this density of states will have any relationship to the energy