The Ising Model

Assignment 4
Prof. Machta
Integrated Workshop

Part 1: Complete the code to perform Metropolis updates of your Ising model. Recall, the energy for the Ising model is as follows:

$$E(s) = -J\sum_{\langle i,j\rangle} s_i s_j - h\sum_j s_j,$$

for a system with constant interaction strength J. $\langle i, j\rangle$ denotes a sum over all nearest neighbors (*i* and *j*). According to the Metropolis algorithm, a trial move is accepted with probability, $p = \mathrm{e}^{-\beta\Delta\mathrm{E}}$.

1. Complete your code, replacing every '???' with appropriate code in the function IsingUpdateHW(). A hint: think carefully about how to implement periodic boundary conditions, noting that Matlab indices go from 1 to N (not 0 to N-1). The "mod" function may be useful.

2. Argue that your algorithm satisfies detailed balance- that is once it is in equilibrium the probability flux from any state A->B is the same as the flux from B-> A

3. Argue that your algorithm is ergodic: that there is some non-zero probability of going from any one state to any other state in finite time. Your argument doesn't need to be too mathematical.

4. (Optional) There are many tricks to speed up this algorithm, most of which aren't worth doing for this assignment. But there is some low hanging fruit! Most of your computing time is currently being spent exponentiating one of 5 numbers over and over again. You will see a large speedup if you instead write these 5 numbers to an array and then call the appropriate entry. (Now all your computing time is spent generating random numbers and accessing random elements of large arrays.)

Part 2: Explore the Ising model qualitatively

For all of this part you may use the code attached (makeFilm) as a template, which will generate a .gif of your Ising code. Modify as required. Use 256x256 lattices.

How does the temperature, t, influence your code?

1. t is rescaled by Onsager's exact Tc. Describe what this means, i.e. how does J/k_BT relate to your inputted t?

2. First explore high t, say t=2. Equilibrate your model from three different starting conditions- every spin +1, every spin -1 and each spin randomly ± 1. To equilibrate, run for at least 100 sweeps (more on this below!) and then make a .gif. Describe and interpret your results.

3. Next explore low t, say t=.5. Again equilibrate your model from three different starting conditions- every spin +1, every spin -1 and each spin randomly ± 1. Describe and interpret your results. Discuss the meaning of equilibration here.

4. Finally, explore the regime near the critical point, t=1. Run simulations just above the critical point at t=1.05 and t=1.1
5. Repeat your simulation with h=.02 and h=.1 for your high and low temperature solutions. Describe your results.

Part III: quantitative exploration of the Ising model:

1. How can you tell if your model is equilibrated? Plot the magnetization (average value of a spin) as a function of number of sweeps for two initial conditions. First, starting from all +1, second from all -1.
    a. Repeat this at t=2,t=1.5,t=1.1,t=1.05.
    b. Repeat this at t=.5, but do not wait for the two curves to approach each other. Explain your results.

2. Calculate the spontaneous magnetization m=<s> as a function of temperature along the line where h=+epsilon. Plot this for 0<t<2, spaced by .1 . Some hints:
    a. h=+epsilon means that when the magnetization is 0 it has no effect, but that when the system is magnetized you always want the states where the magnetization is positive.
    b. To implement this with Monte-carlo, you can always start from a +magnetized state, say all +, and assume the system will never cross.
    c. To reduce noise in your measurement, make sure to average at least 20 configurations for each measurement.

3. Estimate the susceptibility dM/dh. Again plot this for 0<t<2, spaced by .1. To do this, I recommend taking a finite difference derivative by repeating the curve from above, but with h=.02

Part 4 (optional): Kawasaki Algorithm.

In the Metropolis algorithm, the basic move was to (1) choose a random spin, (2) flip it or not so as to satisfy detailed balance. In the Kawasaki algorithm is to (1) choose a random spin, (2) choose one of its 4 neighbors, (3) swap them or not so as to satisfy detailed balance

1. Write IsingUpdateKawasaki() which modifies IsingUpdateHW() to implement the Kawasaki algorithm.
2. Does this algorithm satisfy detailed balance?
3. Is this algorithm ergodic?
4. Make some gifs to compare the Kawasaki and Metropolis algorithms. Describe and discuss the differences.

Submission:
Write and comment the code you used for Part 3 in a file titled main.m. Compress your solutions to a zip (or tar) file containing all of your code in .m format, all your graphs in .png format, and the answers to the questions in .txt, .rtf, or .pdf format. Upload to the canvas Assignments section, named with the format LASTNAME-FIRSTNAME-4.zip (or .tar / .tgz).