Hayley Weir

3rd YEAR CMP COMPULSORY EXPERIMENT

Introduction to the Ising model

TASK:

1. **Show that the lowest possible energy for the Ising model is $E = -DNJ$, where D is the number of dimensions and N is the total number of spins.**
2. **What is the multiplicity of this state?**
3. **Calculate its entropy.**

1.

$$E = -\frac{1}{2}J\sum_{i}^{N}\sum_{j\in neighbours\,(i)} s_i\, s_j$$

In the ground state all spins face the same direction. By trail:

Dimension	$\sum_{j\in neighbours\,(i)} s_i\, s_j$ in ground state
1D	2
2D	4
3D	6

Gives the relationship:

$$\sum_{j\in neighbours\,(i)} s_i\, s_j = 2D$$

$$\sum_{i}^{N}\sum_{j\in neighbours\,(i)} s_i\, s_j = N(2D)$$

Therefore:

$$E = -\frac{1}{2}J\sum_{i}^{N}\sum_{j\in neighbours\,(i)} s_i\, s_j = -\frac{1}{2}J \times N(2D)$$

And so the lowest possible energy is:

$$E = -DNJ$$

2. The multiplicity of this state is 2.
3. Definition of entropy:

$$S = k_B lnW$$

Where W is number of microstates, k_B is the Boltzmann constant and S is the entropy.
Since W = 2:

$$S^0 = 1.38 \times 10^{-23} \times ln2$$
$$S^0 = 9.565 \times 10^{-24}\ \mathrm{JK}^{-1}$$

TASK: Imagine that the system is in the lowest energy configuration. To move to a different state, one of the spins must spontaneously change direction ("flip").

1. **What is the change in energy if this happens ($D = 3,\ N = 1000$)?**
2. **How much entropy does the system gain by doing so?**

Hayley Weir

1. Lowest energy configuration is all spins aligned.

1 spin flipped:

$$E = -\frac{1}{2}J((2ND) + 6 + 6)$$
$$E = -J(ND + 6)$$

because loss of favourable interaction gives +6, and gain of unfavourable interaction gives +6 again since ↓↑ is unfavourable. The number 6 is due to each cell having 6 adjacent cells in a 3-D structure and so each 6 * -1 = -6 as the cell has opposite spin to the adjacent cells.

When D = 3 and N = 1000:

$$E = -J * (3 * 1000 + 6)$$
$$E = -2994J$$

Change in E between ground state and spin flipped state:

$$\Delta E = E - E^0$$
$$\Delta E = -2994J - -3000J = +\boldsymbol{6J}$$

2. The entropy becomes:

W = N = 1000

$$S = 1.38E - 23 * ln1000 = 9.533E - 23 \ \ \mathrm{JK^{-1}}$$

Therefore the change in entropy is:

$$\Delta S = S - S^0$$
$$\Delta S = 9.533E - 23 - 9.565E - 24 = 8.576E - 23\ \mathrm{JK^{-1}}$$
$$\boldsymbol{\Delta S = 51.63\ \mathrm{JK^{-1}mol^{-1}}}$$

Hence the system has increased in entropy as it has become more disordered.

TASK:

1. **Calculate the magnetisation of the 1D and 2D lattices in figure 1.**
2. **What magnetisation would you expect to observe for an Ising lattice with** $D = 3,\ N = 1000$ **at absolute zero?**

1. Expression for magnetization is given by:

$M = \sum_i s_i$

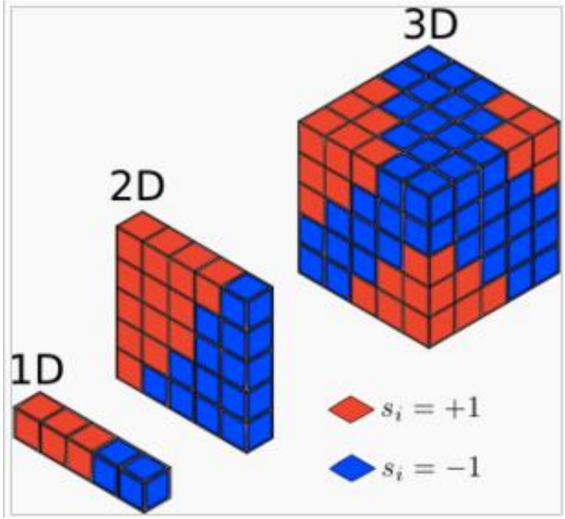
1D lattice: M = 3 - 2 = **1**

2D lattice: M = 13 – 12 = -**1**

2. At 0 K the system will be in its ground state. Therefore all the spins will be in the same directions (all +1 or all -1).

D = 3, N = 1000

M = ±1000

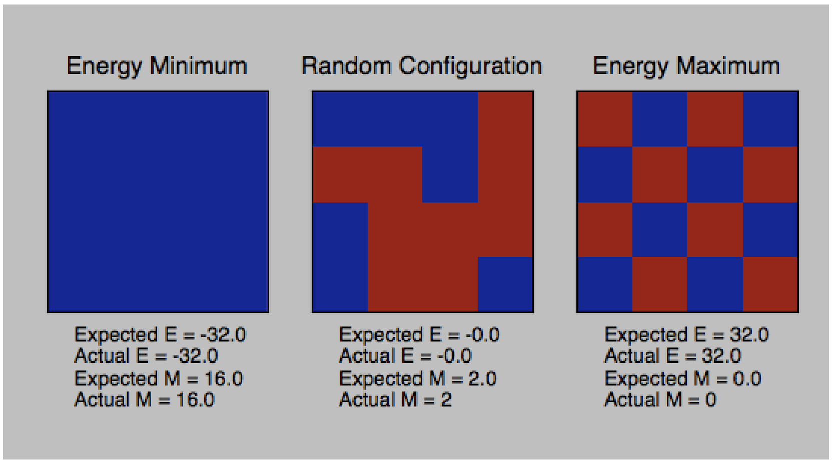


Calculating the energy and magnetisation

TASK: complete the functions energy() and magnetisation(), which should return the energy of the lattice and the total magnetisation, respectively. In the energy() function you may assume that $J = 1.0$ at all times (in fact, we are working in *reduced units* in which $J = k_B$, but there will be more information about this in later sections). Do not worry about the efficiency of the code at the moment — we will address the speed in a later part of the experiment.

EnergyAndMagnetisation.py

TASK: Run the ILcheck.py script from the IPython Qt console using the command



```
Expected E = -32.0
Actual E = -32.0
Expected M = 16.0
Actual M = 16.0

Expected E = -0.0
Actual E = -0.0
Expected M = 2.0
Actual M = 2

Expected E = 32.0
Actual E = 32.0
Expected M = 0.0
Actual M = 0
```

Introduction to Monte Carlo simulation

TASK:

1. **How many configurations are available to a system with 100 spins?**
2. **To evaluate these expressions, we have to calculate the energy and magnetisation for each of these configurations, then perform the sum. Let's be very, very, generous, and say that we can analyse 1×10^9 configurations per second with our computer. How long will it take to evaluate a single value of $\langle M \rangle_T$?**

1. Since there are 2 options of what each cell could be (+1 or -1):

$$\text{no. of possible configs} = 2^{100} = \mathbf{1.26} \times 10^{30}$$

Time to evaluate configurations of 100 spin system, T:

$$\mathrm{T} = \frac{1.26 \times 10^{30}}{1 \times 10^{9}} = 1.268 \times 10^{21} \text{ seconds}$$
$$\mathbf{T = 40,181,353,532,343 \ years}$$

TASK: Implement a single cycle of the above algorithm in the montecarlocycle(T) function. This function should return the energy of your lattice and the magnetisation at the end of the cycle. You may assume that the energy returned by your energy() function is in units of k_B! Complete the statistics() function.

This should return the following quantities whenever it is called: $< E >, < E^2 >, < M >, < M^2 >$ **, and the number of Monte Carlo steps that have elapsed.**

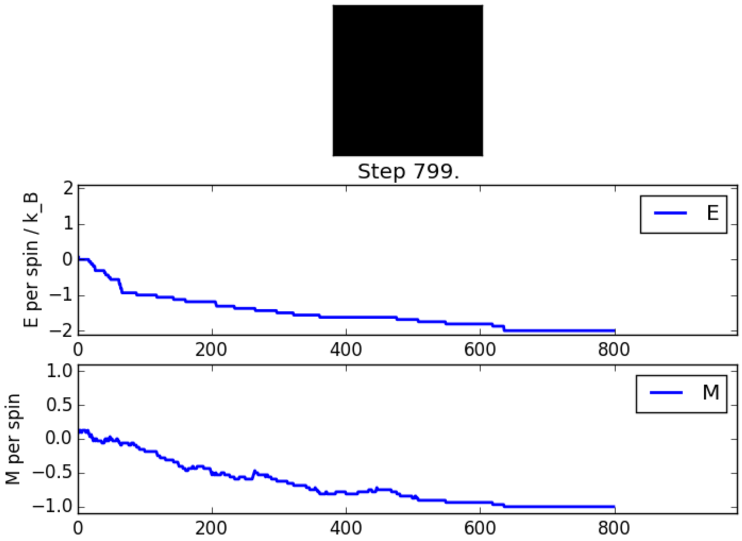
MonteCarloAndStastics.py

TASK:

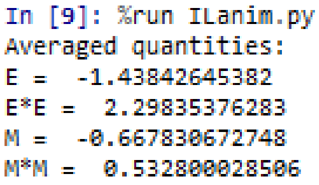
1. **If** $T < T_C$ **, do you expect a spontaneous magnetisation (i.e. do you expect** $\langle M \rangle \neq 0$**)?**
2. **When the state of the simulation appears to stop changing (when you have reached an equilibrium state), use the controls to export the output to PNG and attach this to your report.**
3. **You should also include the output from your statistics() function.**

1. If Temp is less than the Currie temperature then expect <M> ≠0 as all the spins will be aligned in a ferromagnetic material so there will be a strong dipole.

2.



3.



```
In [9]: %run ILanim.py
Averaged quantities:
E =  -1.43842645382
E*E =  2.29835376283
M =  -0.667830672748
M*M =  0.532800028506
```

Accelerating the code

TASK: Use the script ILtimetrial.py to record how long your *current* version of IsingLattice.py takes to perform 2000 Monte Carlo steps. This will vary, depending on what else the computer happens to be doing, so perform repeats and report the error in your average!

Time to perform 2000 steps/ seconds
8.111257549
8.084071129
8.090109769

8.047899733
8.036258632
8.047862672
8.083449932
Standard Deviation: 0.02764491
Standard Error: 0.010448794
Average: **8.071558488 ± 0.010448794**

TASK: Look at the documentation for the NumPy sum function. You should be able to modify your magnetisation() function so that it uses this to evaluate M. The energy is a little trickier. Familiarise yourself with the NumPy roll and multiply functions, and use these to replace your energy double loop (you will need to call roll and multiply twice!).

AcceleratedEnergyAndMagnetisation.py:

TASK: Use the script ILtimetrial.py to record how long your *new* version of IsingLattice.py takes to perform 2000 Monte Carlo steps. This will vary, depending on what else the computer happens to be doing, so perform repeats and report the error in your average!

Accelerated Code
0.182258169
0.190376778
0.179950703
0.179642813
0.180751218
0.208782055
0.17884857
Standard Deviation: 0.010859743
Standard Error: 0.004104597
Average: **0.185801472 ± 0.004104597**

The effect of temperature

TASK: The script ILfinalframe.py runs for a given number of cycles at a given temperature, then plots a depiction of the *final* lattice state as well as graphs of the energy and magnetisation as a function of cycle number. This is much quicker than animating every frame! Experiment with different temperature and lattice sizes. How many cycles are typically needed for the system to go from its random starting position to the equilibrium state? Modify your statistics() and montecarlostep() functions so that the first N cycles of the simulation are ignored when calculating the averages. You should state in your report what period you chose to ignore, and include graphs from ILfinalframe.py to illustrate your motivation in choosing this figure.

Steps to reach equilibrium
450
520
800

2800
400
890
550

Maximum was 2800 so 4000 was used as it is significantly bigger than 2800, however it isn't so big so that it takes a lot of time to get results.

Calculating the heat capacity

TASK: By definition, $C = \frac{\partial \langle E \rangle}{\partial T}$ **From this, show that** $C = \frac{\mathrm{Var}[E]}{k_B T^2}$.

$$C = \frac{\partial \langle E \rangle}{\partial T}$$

$$[E] = \frac{1}{Z}\sum_{i}^{\infty} E_i e^{-\frac{E_i}{k_B T}} \qquad (1)$$

Z – Normalisation factor so when integrated over all space equals 1.

$$Z = \sum_{i}^{\infty} e^{-\frac{E_i}{k_B T}} \qquad (2)$$

combining (1) and (2) gives:

$$[E] = \frac{1}{\sum_i^\infty e^{-\frac{E_i}{k_B T}}} \sum_{i}^{\infty} E_i e^{-\frac{E_i}{k_B T}}$$

therefore:

$$C = \frac{\partial \langle E \rangle}{\partial T} = \frac{\partial}{\partial T}\left(\frac{\sum_i^\infty E_i e^{-\frac{E_i}{k_B T}}}{\sum_i^\infty e^{-\frac{E_i}{k_B T}}}\right)$$

Substituting

$$A = \sum_{i}^{\infty} E_i e^{-\frac{E_i}{k_B T}}$$

and

$$Z = \sum_{i}^{\infty} e^{-\frac{E_i}{k_B T}}$$

$$\frac{\partial \langle E \rangle}{\partial T} = \frac{\partial}{\partial T}\left(\frac{A}{Z}\right) = -\frac{A}{Z^2}\frac{\partial Z}{\partial T} + \frac{1}{Z}\frac{\partial A}{\partial T}$$

$$\frac{\partial Z}{\partial T} = \sum_{i}^{\infty} \frac{E_i}{k_B T^2} e^{-\frac{E_i}{k_B T}}$$

$$\frac{\partial A}{\partial T} = \sum_{i}^{\infty} \frac{E_i^2}{k_B T^2} e^{-\frac{E_i}{k_B T}}$$

Therefore, substituting gives:

$$\frac{\partial\langle E\rangle}{\partial T}=\frac{\partial}{\partial T}\left(\frac{A}{Z}\right)=-\frac{A}{Z^2}\sum_{i}^{\infty}\frac{E_i}{k_BT^2}e^{-\frac{E_i}{k_BT}}+\frac{1}{Z}\sum_{i}^{\infty}\frac{E_i^2}{k_BT^2}e^{-\frac{E_i}{k_BT}}$$

$$=\frac{1}{k_BT^2}\left(-\frac{A}{Z}\sum_{i}^{\infty}E_ie^{-\frac{E_i}{k_BT}}+\frac{1}{Z}\sum_{i}^{\infty}E_i^2e^{-\frac{E_i}{k_BT}}\right)$$

$$=\frac{1}{k_BT^2}\left(-\frac{A^2}{Z^2}+\sum_{i}^{\infty}\frac{1}{Z}E_i^2e^{-\frac{E_i}{k_BT}}\right)$$

$$=\frac{1}{k_BT^2}(-\langle E\rangle^2+\langle E^2\rangle)$$

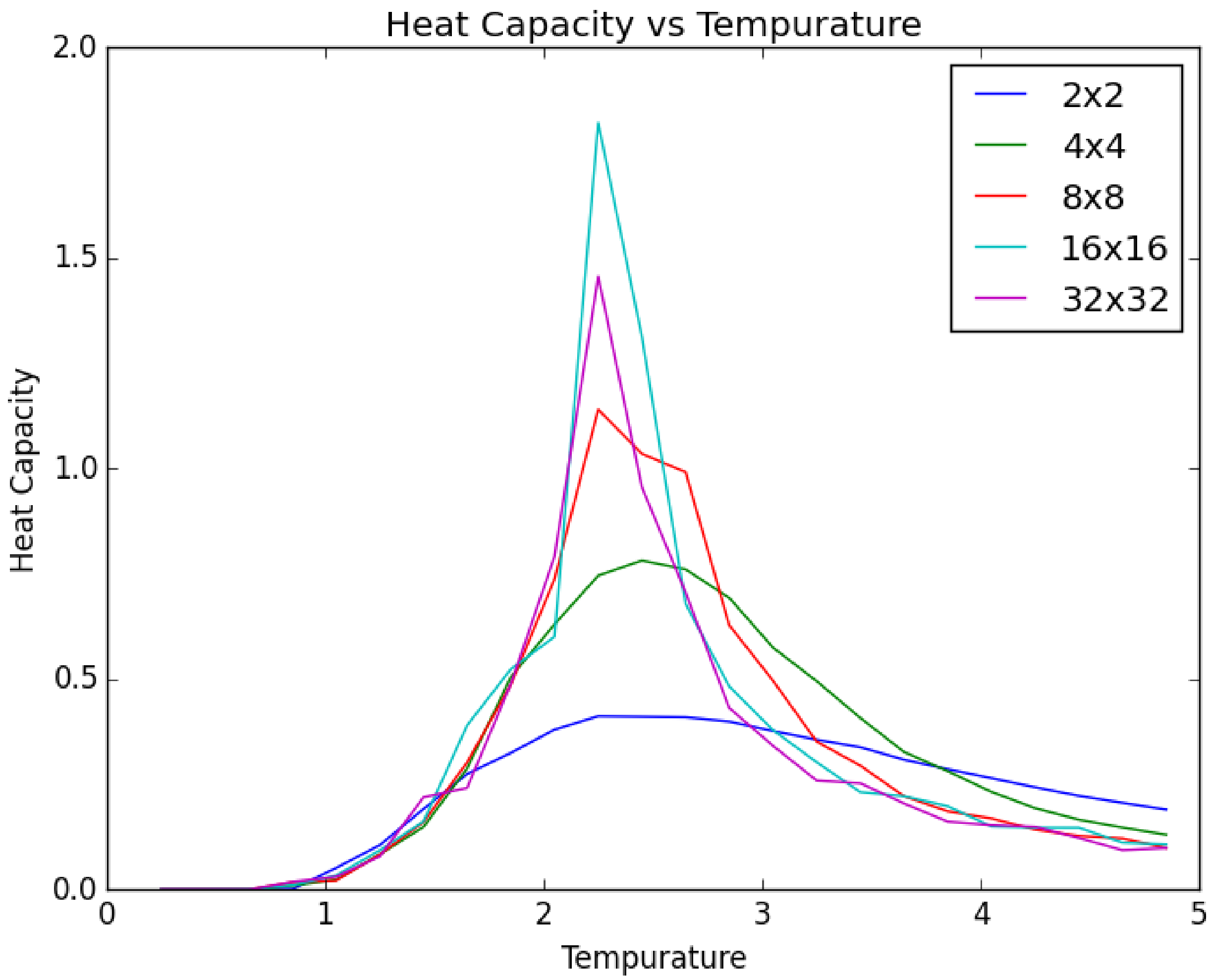
Since:

$$\mathrm{Var}[E]=\langle E^2\rangle-\langle E\rangle^2$$

Gives the relationship:

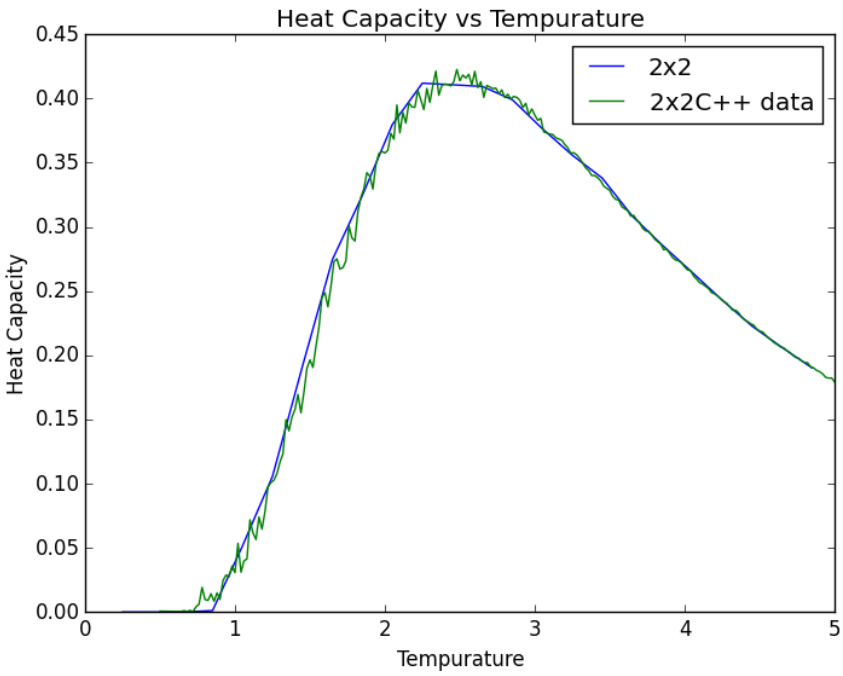
$$\frac{\partial\langle E\rangle}{\partial T}=\frac{\mathrm{Var}[E]}{k_BT}$$

TASK: Write a Python script to make a plot showing the heat capacity versus temperature for each of your lattice sizes from the previous section. You may need to do some research to recall the connection between the variance of a variable, $\mathrm{Var}[X]$, the mean of its square $\langle X^2\rangle$, and its squared



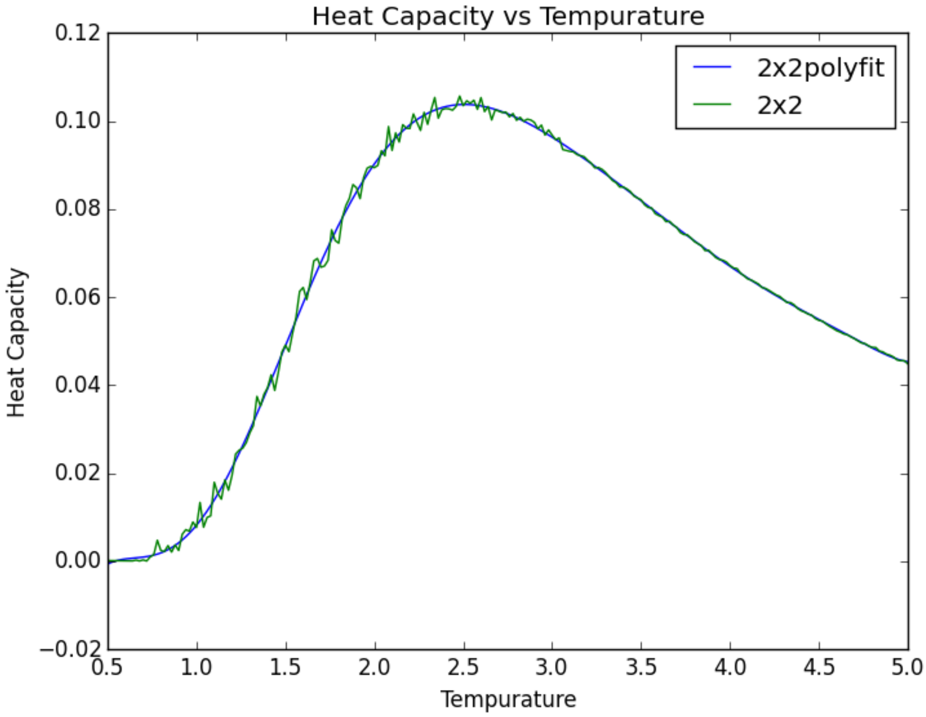
mean $\langle X\rangle^2$. You may find that the data around the peak is very noisy — this is normal, and is a result of being in the critical region. As before, use the plot controls to save your a PNG image of your plot and attach this to the report.

TASK: A C++ program has been used to run some much longer simulations than would be possible on the college computers in Python. You can view its source code here if you are interested. Each file contains six columns: T, E, E^2, M, M^2, C (the final five quantities are per spin), and you can read them with the NumPy loadtxt function as before. For each lattice size, plot the C++ data against your data.
For *one* lattice size, save a PNG of this comparison and add it to your report — add a legend to the graph to label which is which. To do this, you will need to pass the label="..." keyword to the plot function, then call the legend() function of the axis object (documentation here).



TASK: write a script to read the data from a particular file, and plot C vs T, as well as a fitted polynomial. Try changing the degree of the polynomial to improve the fit — in general, it might be difficult to get a good fit! Attach a PNG of an example fit to your report.

CPlusPlusVsMyDataPolyfit.py



TASK: Modify your script from the previous section. You should still plot the whole temperature range, but fit the polynomial only to the peak of the heat capacity! You should find it easier to get a good fit when restricted to this region.

CPlusPlusVsMyDataPolyfitPart2.py

TASK:

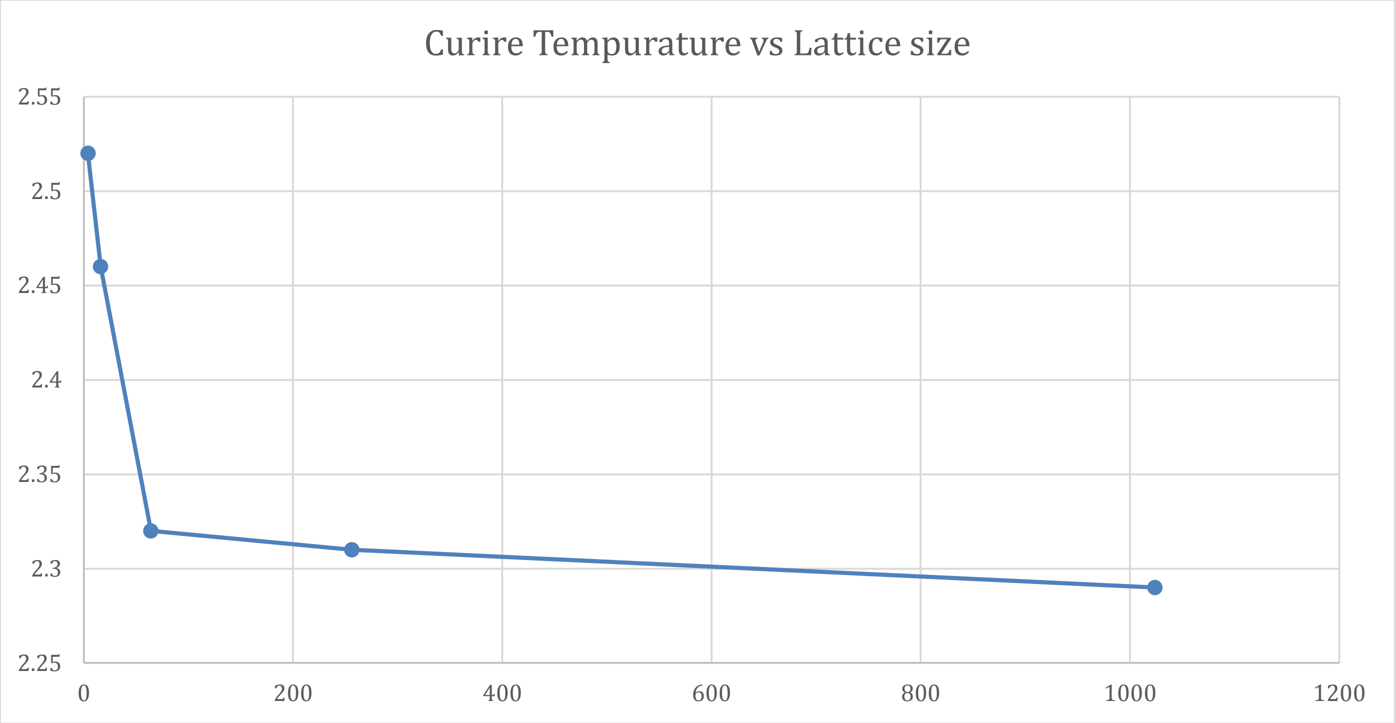
1. **Find the temperature at which the maximum in C occurs for each datafile that you were given.**
2. **Make a text file containing two colums: the lattice side length (2,4,8, etc.), and the temperature at which C is a maximum. This is your estimate of T_C for that side length.**
3. **Make a plot that uses the scaling relation given above to determine $T_{C,\infty}$. By doing a little research online, you should be able to find the theoretical exact Curie temperature for the infinite 2D Ising lattice.**
4. **How does your value compare to this?**
5. **Are you surprised by how good/bad the agreement is?**
6. **Attach a PNG of this final graph to your report, and discuss briefly what you think the major sources of error are in your estimate.**

1.

2.

Lattice Size	Max Temperature/K (T_C)
2x2	2.52
4x4	2.46
8x8	2.32
16x16	2.31
32x32	2.29

3.



The graph reaches an asymptote at ~**2.275**. This is determined by observation as no trend lines match this graph well.

Reference for duality argument: 1941, H.A. Kramers and G.H. Wannier, *Phys. Rev.* **60**, 252 (1941)

$$\frac{k_B T_C}{J} = \frac{2}{\log(1+\sqrt{2})} = 2.269 \ldots$$

This is very similar to the Curie temperature giving an error of 2.275 ± 0.006, which is much smaller than I was expecting!

The main sources of error in this experiment are:

1. The choice of 4000 as the cut off limit for the number of steps before taking the average.
2.