Project 2: Ising Model

Rohan Mehra: 640052491 *PHYM004* *Submitted: 20/01/15*

Contents

Program Summary:

This program simulates ferromagnetism in two spatial dimensions using the statistical mechanics approach of the Ising Model. An array of randomly distributed spins is generated, and the Metropolis algorithm is used to evolve the system in time until equilibrium is reached (macroscopic properties of the system are constant). At equilibrium the macroscopic properties of System Energy, Magnetisation, Specific Heat Capacity and Susceptibility are calculated. The system is evolved through a range of temperatures and the macroscopic properties at equilibrium for each temperature are printed to file. The source code must be compiled according to C99 standards.

The program must be called with one command line argument, the output data filename.

./Ising_sim [-opt] <output_filename>

The program can also be called with any combination of 7 options:

Options:

-m: Set the maximum number of iterations for equilibrium to be reached at a given temperature. Default: 1,000,000,000.

-t: Set equilibrium threshold. Equilibrium is found once the relative change in energy is no more than this value ever N time steps, where N is the number of spins in the lattice (N = DxD). Default: 1e-4

-i: Set initial temperature. The maximum and initial temperature of the system in units of J/kb. Default: 3.5
-f: Set final temperature, not inclusive. Final and minimum temperature will be final_temp + temp_step_size Default: 0
-s: Set temperature step size as the system is cooled. Default 0.1
-d: Set the data record multiple. This number determines for how long energy and magnetisation values are recorded and averaged to calculate the macroscopic properties. A value of 2 means macroscopic quantities are calculated by averaging values for the same amount of time it took to get to equilibrium. A value of 3 would mean values are averaged for twice the time it took to get to equilibrium, *etc.*. Default: 2
-a: Enable *state_writer()* with ASCII art. This option turns on *state_writer()* before and after the system is temperature evolved. This prints out the spin state of each particle and represents this in an 'ascii art' format to visualise domains.
Note: On Mac operating systems options must be given before command line arguments. On Linux and Windows order is irrelevant.
Bug: If equilibrium time takes longer than *max_steps/2,* the simulation will only record data up to *max_steps irrespective* of the *data_record_multiple* without telling the user. This is easily fixable in future versions.

The Ising Model and Metropolis Algorithm:

The Ising model is the simplest theoretical model of ferromagnetism which exhibits a spontaneous magnetisation phase change consistent with observations. In particular, the two dimensional model consist of an infinite lattice of discrete variables which represent atomic spins. These spins can be either spin UP (+1) or spin down (-1) along the z-axis and interact with their immediate neighbouring particles via an 'exchange' interaction. The energy of the finite section of the lattice is characterised by a term due to nearest neighbour spin interaction and interaction with an external magnetic field **B** and is given by:

$$E = -\frac{J}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\mathrm{nn}(i,j)\mathbf{s}_i\cdot\mathbf{s}_j\right)-\mu_0\sum_{i=1}^{N}\left(\mathbf{B}\cdot\mathbf{s}_i\right)$$

Where $nn(i,j) = 1$ if s_i and s_j are nearest neighbours, and 0 otherwise.
In my simulation, the Ising model is simulated by an array of doubles with periodic boundary conditions and is evolved using a Monte Carlo Metropolis Algorithm. The algorithm flips a certain spin and computes the change in energy for the system. If the energy of the system reduces the flip is accepted. Otherwise, a random number is generated and if this number is less than a Boltzmann distribution at the current temperature the spin flip is also accepted, simulating thermodynamic excitations of the system. This is repeated until equilibrium is reached, when the energy of the system is minimized.

Implementation:

Code was designed with the top priorities of modularity, readability and expandability; followed by speed and efficiency considerations.

Primary data structures:

There are five primary data structures in the program.
The first is the 2D array *double S[D][D]* which represents the lattice of spins. This array is accessed using the macro *S(x,y) [(x)%D][(y)%D]* to simulate an infinite lattice and results in periodic boundary conditions. It is passed to other functions by reference (like all C arrays) and hence is modified by these functions.

The second is the *flip_probs[2*FP_OFFSET +1]* array. In the Metropolis algorithm a spin is flipped if a randomly generated number falls inside a Boltzmann probability distribution). This distribution depends on the sum of the adjacent spins of the particle of interest, temperature and the value of the current spin, and requires the computation of an exponential. As the sum of the adjacent spins can only take 9 distinct values it is computationally advantageous to calculate the values of the distribution (not including the value of the current spin) at each temperature before hand, rather than computing an exponential at each time step. This array stores the 9 possible values for the Boltzmann probability distribution up to a sign in the exponential and is repopulated at every temperature, saving computational resources required to compute millions of exponentials. These computed values are reciprocated if the spin of interest is negative.

The final three are structures which pass arguments to other functions and **are primarily to keep code modular.**
The *struct options* contains all the values for options which may be user inputted at invocation.
The *struct EM_sums* contains the values calculated in each invocation of the *spin_flip_Metrop* function which are required for calculation of macroscopic properties.
The *struct properties* contains all calculated macroscopic properties for a given temperature which need to be printed.
All structures are passed by reference and are initialised outside of loops to avoid repeated memory allocation and the potential for memory leaks.

Main:

The *main* function serves three primary purposes. Firstly, it initialises the 2D array *S[D][D]* which represents the lattice of spins which make up the system. This array is then passed to *rand_pop_S* which sets each element of the array to a random spin orientation as a starting point for the material. Secondly, it processes arguments and options given by the user and initialises an instance of *struct options opts*. The members of *opts* are set to the default values at initialisation, which are pre-processor defined, and modified as required using the *getopt* function. This structure is passed to other functions by reference so that the options are made available throughout the program. Finally, *main* calls the *temperature_evolve* function which modifies *S* and calculates and prints properties of interest.

Temperature Evolve:

The *temperature_evolve* function initialises instances of *flip_probs* array, *struct EM_sums EM_sums1* and *struct properties props* and opens the output file for writing. The function then loops through the various temperatures, and repopulates the *flip_prob* array for each temperature. The Metropolis algorithm then finds equilibrium and properties are calculated and printed at each temperature. The *structs* are passed by reference to prevent memory from being repeatedly allocated.

Metropolis Evolve:

The Metropolis algorithm steps through every spin particle sequentially and flips a spin if it is energetically favourable or thermodynamically probable to do so. Going through the array sequentially results in faster equilibrium times, but does not accurately evolve the system in time [1]. As we are only interested in equilibrium properties, however, it is computationally advantages to do so. A spin will flip if $\Delta E_{sys} \leq 0$ or if a randomly generated number between 0 and 1 is less than a Boltzmann distribution $e^{-\frac{\Delta E}{k_B T}}$. If $\Delta E_{sys} \leq 0$ then $e^{-\frac{\Delta E}{k_B T}} \geq 1$ and hence will always result in flipped spin, so only this condition needs to be tested [1]. As the sign of ΔE depends on the current spin of the particle of interest the pre-calculated exponentials in *flip_probs* may need to be reciprocated depending on the sign of the spin. At the beginning of the algorithm the total energy and magnetisation are calculated, and these values are modified by ΔE and ΔM if a spin is flipped. The function repeats this process until equilibrium is reached.

In this program the Metropolis algorithm attempts to minimise the energy of the system and is a form of simulated annealing algorithm. Hence, I decided the energy would be the best macroscopic property to observe to decide when equilibrium is reached, in addition to it being much easier to calculate than specific heat or susceptibility. From early investigations it was clear that the energy of the system followed a general exponential decay similar to that of a cooling object. The program takes samples of the energy every *N* time steps and computes the relative change compared to the last sample (the relative change is measure with respect to the most recent energy as this prevents deceptively small changes from being generated due to asymptotically large previous energies). If this is below some manually supplied threshold value, then the boolean *isEquilibrium* is set to *true.* To ensure that equilibrium is not found erroneously due to noise, the program must find equilibrium twice in a row. This is achieved through relatively straight forward logic and a second Boolean variable *confirm_equilibrium*.

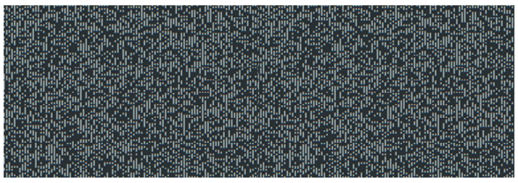
Once equilibrium is achieved macroscopic values must be calculated. The heat capacity and susceptibility calculations require calculating the time variance of the energy and magnetisation respectively at equilibrium. Standard variance calculations can be numerically unstable and cause 'catastrophic cancellation' and loss of precision [2]. To avoid this, a shifting algorithm was used to keep the variance unchanged, but retain more precision. When equilibrium is reached the value for the energy and magnetisation are stored as *energy_shift* and *mag_shift*. These values are subtracted from all energy values used in the variance calculation to reduce the magnitude of the energy and magnetisation, resulting on greater retained precision [2]. These values are cumulatively summed and stored in the *EM_sums1* struct for each temperature.

Properties Calculator:

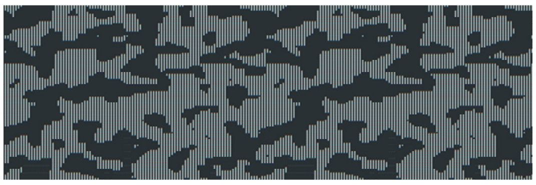
The *properties_calc* function takes a pointer to the *EM_sums* struct and calls the various property calculators and saves these values in the *props* struct. The properties are calculated with the relations (C3 – 6) given in the assignment sheet. All properties are calculated in units where J and k_B are both 1.

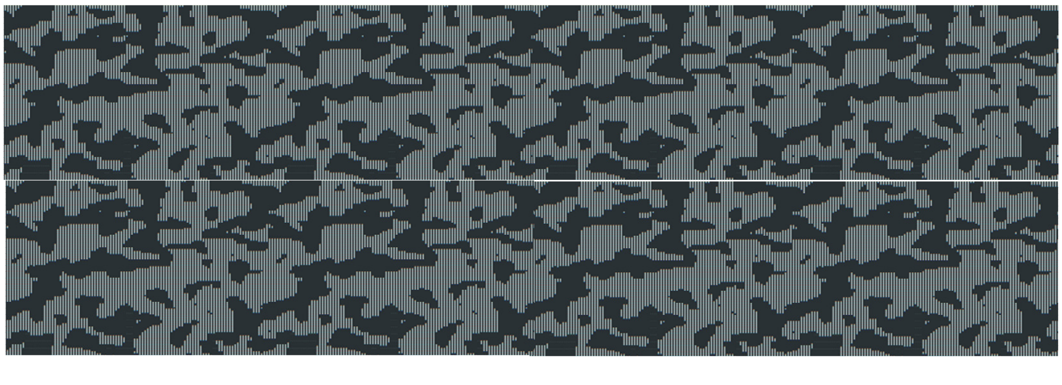
File Writers:

There are two different file writer functions *properties_writer* and *state_writer*. The first, *properties_writer,* simply writes the physical properties to file in tab separated columns for each temperature. *State_writer,* however, prints 2 files – the first is a list of coordinates of S and the spin at these points, while the second is an 'ascii art' representation of the system. The 'ascii art' writer represents the system by printing a *+* for each spin UP particle and leaving a blank space for each spin DOWN particle. This visualisation was quick and easy to code and is a good way to observe equilibrium occurring, as well as the periodic nature of the infinite lattice. If we zoom out in our text editor and take a screenshot we get a good overall picture.



⇨





Above: After approximately 100,000 time steps we can see the system go from a random distribution of spins to ordered domains. We can see the periodic nature of the boundary conditions as the image is repeated to form a continuous image.

Results:

Theoretical Results:

The 2D Ising model has an exact analytic solution derived by Onsager in 1944. The model predicts a phase transition to occur at a critical temperature **Tc**. At this temperature the system undergoes an order to disorder transition and spontaneous magnetisation falls to zero. For an infinite lattice with k and J set to 1, this temperature should occur at [3]:

$$T_c = \frac{2\,J}{k\ln\left(1+\sqrt{2}\right)} = \frac{2}{\ln\left(1+\sqrt{2}\right)} \approx 2.269$$

Additionally, the magnetisation should follow the relation [4]:

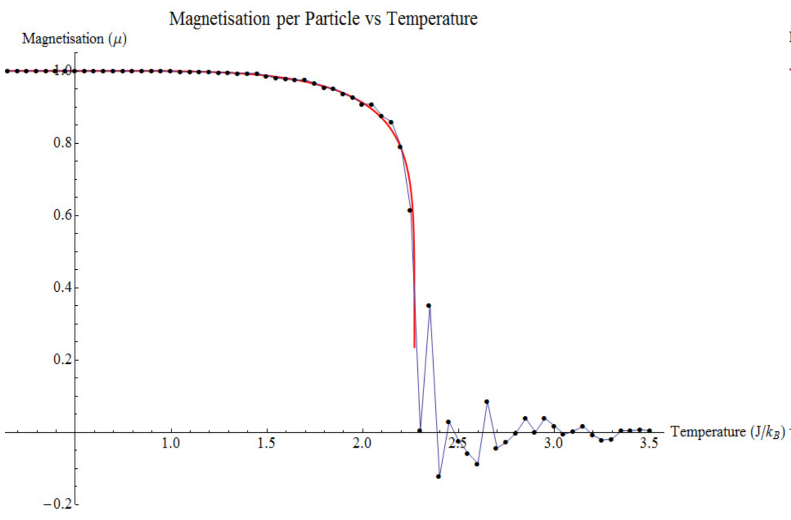
$$M(T) = \left(1 - Sinh\left(\frac{2}{T}\right)^{-4}\right)^{\frac{1}{8}}$$

We should see the magnetisation of the sample immediately go to zero at the critical temperature. We should also see a sharp step in the energy at the critical temperature, as the spins go from a spin aligned, minimum ground state energy, to a completely disordered maximum energy above the critical temperature. The heat capacity is the temperature derivative of the mean energy; hence we expect the heat capacity to have a sharp spike and diverge to infinity at the critical temperature and be zero elsewhere. Finally, the magnetic susceptibility measures how much the magnetisation changes with respect to temperature [5], hence we expect a similar shape to the heat capacity, with a divergence to infinity at the critical temperature and a value of zero elsewhere.

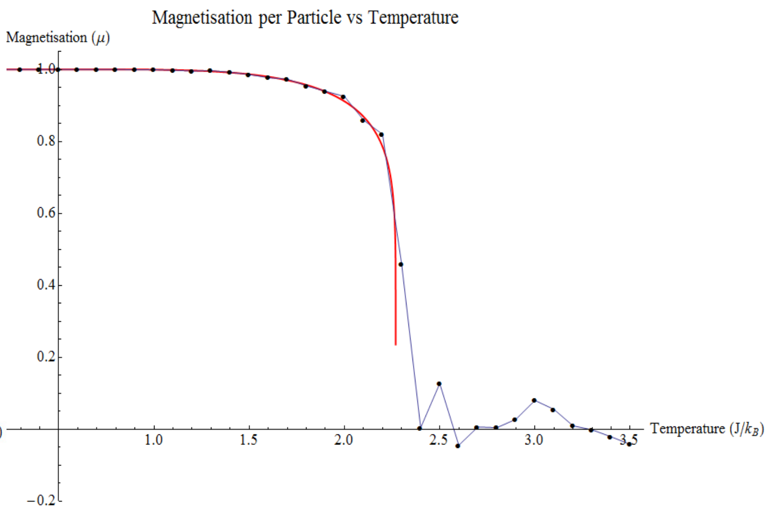
Computed Results:

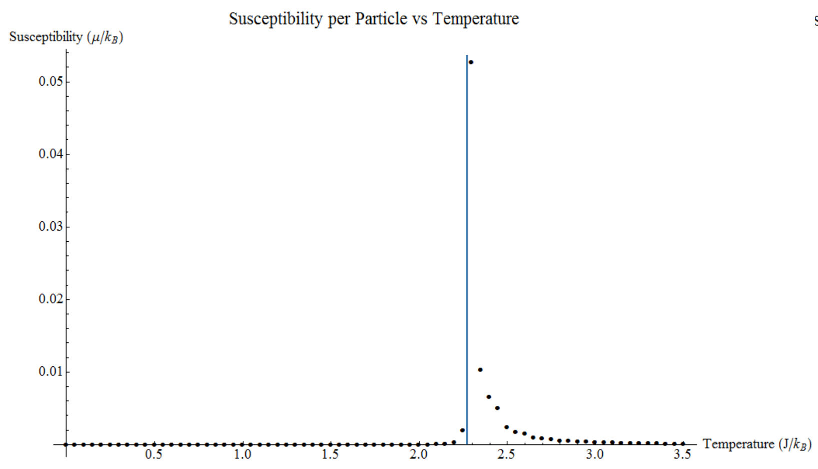
With default values the program took approximately 5-7 minutes to complete inside a virtual machine on my laptop, depending on the random seed. Speed is greatly increased if the lattice size is reduced to D = 50 with a moderate hit to quality of results. The graphs produced below are in very good agreement with other simulations of the model, such as the simulation by Witthauer and Dieterle [5]. Note, however, as there is no external magnetic field the direction of magnetisation is arbitrary, as a result magnetisation may be -1 instead of 1. This can be fixed by negating the results.

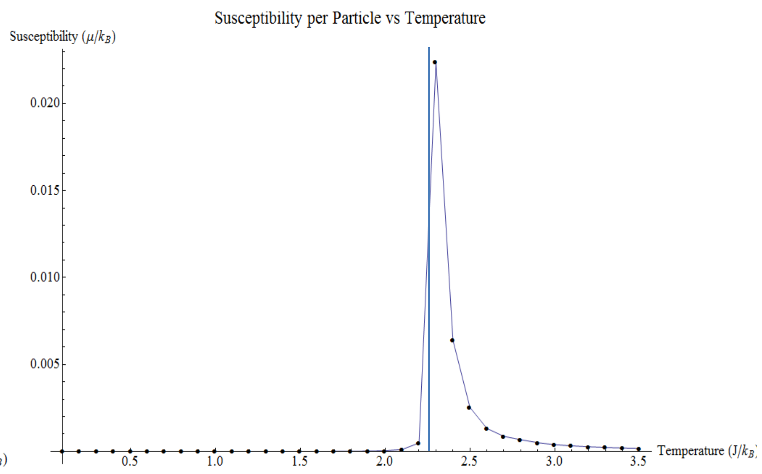
Temperature Step size = 0.05

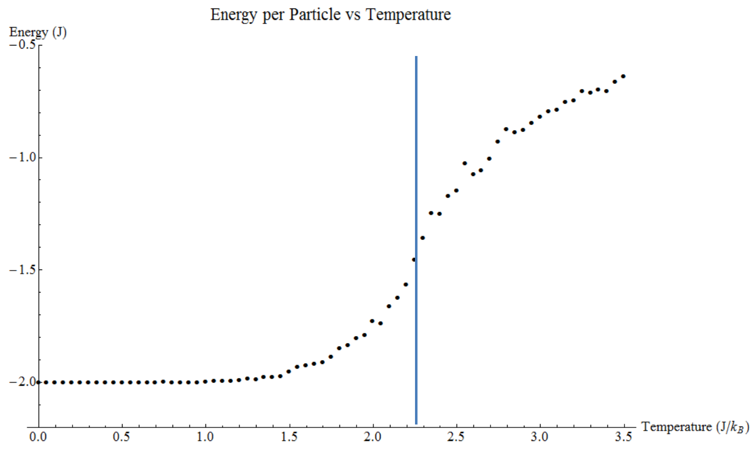


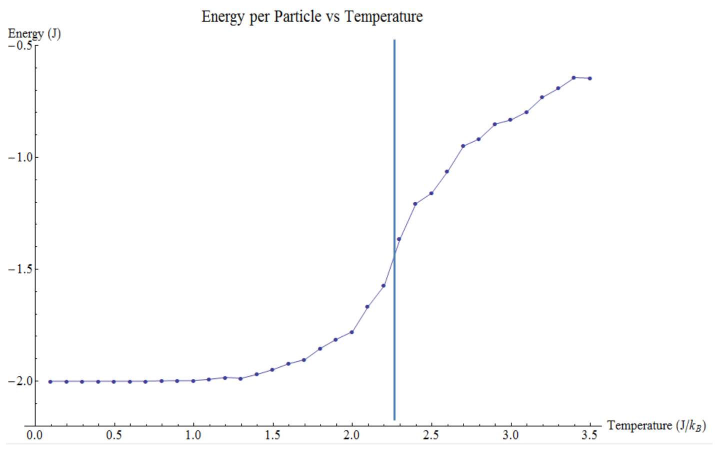
Temperature Step size = 0.1

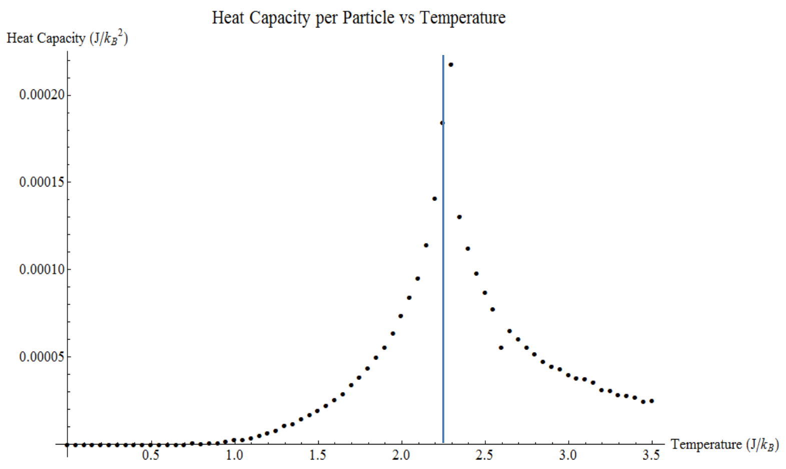


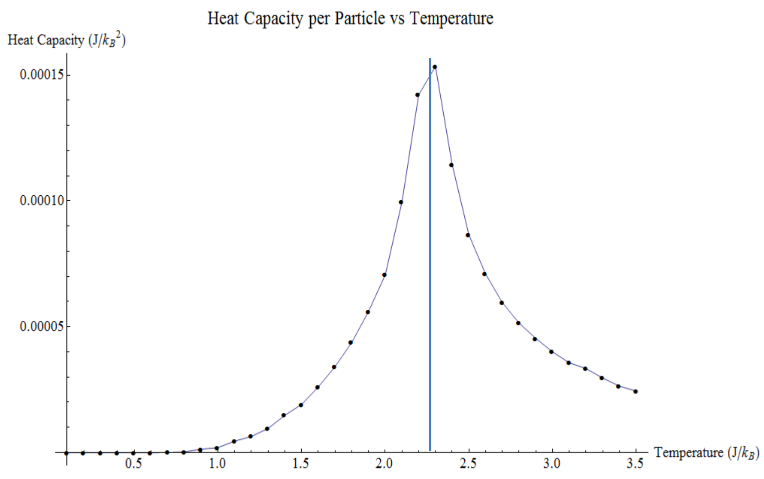












Observing the results for magnetisation with the overlaid theoretical model (red) we can see a very good agreement between the simulation and the expected results. A phase transition is clearly evident, at a critical temperature almost exactly at theoretical value of Tc = 2.269 - with results easily correct to 2 significant figures. There are two clear differences from theoretical values expected, however. Firstly, magnetisation does not immediately drop to a value of zero, but instead shows decreasing amplitude of random noise around 0 after Tc. This can be explained as following. Below the critical temperature spontaneous, global spin alignment occurs, but above the critical temperature local spin alignment (clumping) can still arise, causing residual magnetisation [6]. As the temperature gets higher these residual magnetisations become smaller as thermal excitation randomly distributes the spins. This inconsistency may be due to the finite, periodic nature of the simulation lattice, resulting in local clumping not completely cancelling out as it would on an infinite plane. The second difference is the stretched phase transition. This may be due to temperature step resolution limitations and residual magnetisation acting in the opposite direction to the phase transition, stretching the transition.

This residual magnetisation and stretched phase transition results in the susceptibility not being a perfectly divergent spike at the critical temperature. The smoothening to the right of the susceptibility curve is most likely due to the residual magnetisation which is a result of the finite, periodic nature of the lattice [7]. Regardless, the results are very close to expected, with a sharp spike evident at the critical temperature.

The energy plots are similar to the expected step shape, but are smoothed out and stretched, most likely due to the finite nature of the simulation lattice. An infinite lattice will be completely disordered, but a finite section of lattice may have locally aligned spins resulting in a finite section appearing ordered. There is, however, a clear inflection point at the critical temperature.

The stretched nature of the energy plot results in the heat capacity not being a perfect spike but rather a curve. This result is expected and well documented, with larger lattice sizes theoretically having sharper peaks and smaller lattice sizes being smoother [7].

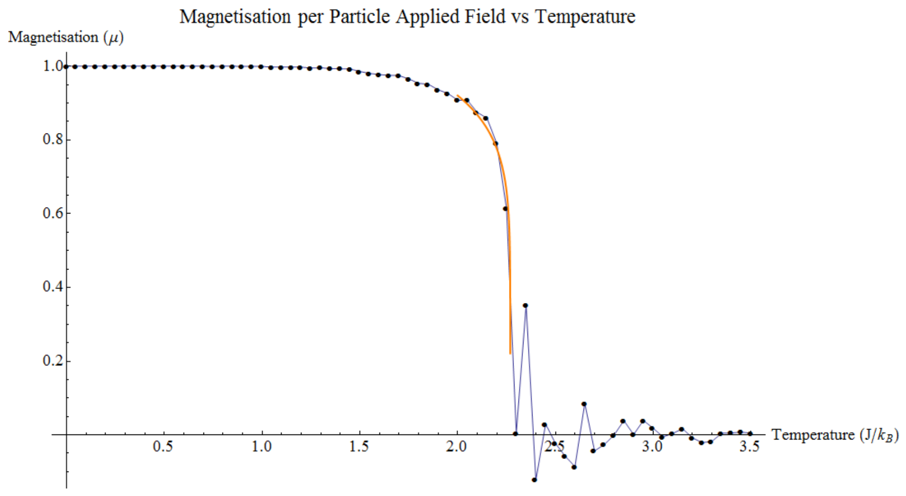
Changing Parameters:

Larger sizes for the lattice size, *D*, result in better, less noisy results but significantly longer execution times, with one lecture series suggesting execution time was proportional to D^4 [8]. A lattice size of 50 provides reasonable data with execution times of approximately a minute. Additionally, reducing the threshold value to less than 1e-4 or reducing the *data_record_multiple* to less than 2, results in poor quality data.

Critical Exponents:

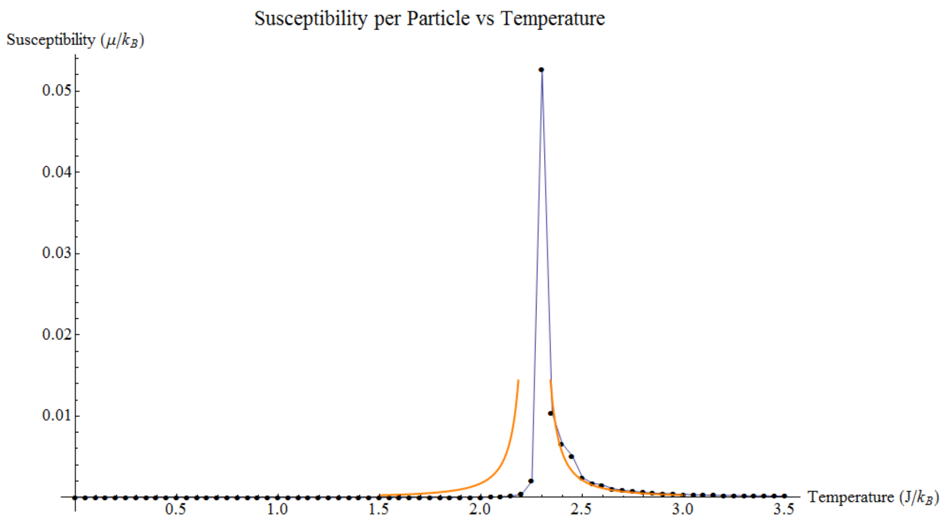
Critical exponents are used to characterise the behaviour of second order phase transitions very close to the critical temperature. Taking a power series about the critical temperature and neglecting high order terms we find that the macroscopic properties follow the general relation $X \propto |T - T_c|^{\lambda}$ close to the critical temperature [7]. Theoretically, the critical exponents should be λ = β = 1/8 for magnetisation, λ = γ = -7/4 for susceptibility and λ = α = 0 for heat capacity. Fitting the data for the exponent failed numerous times on Mathematica, so instead I fit for the constant of proportionality to show that the results fit the theoretical relations.

Fitting the 6 data points before Tc (T = 1.95 → 2.25) for magnetisation to $M = \kappa\,(2.269 - T)^{\frac{1}{8}}$, gave the below plot in orange, showing good agreement with theory, **with κ = 1.0845.**



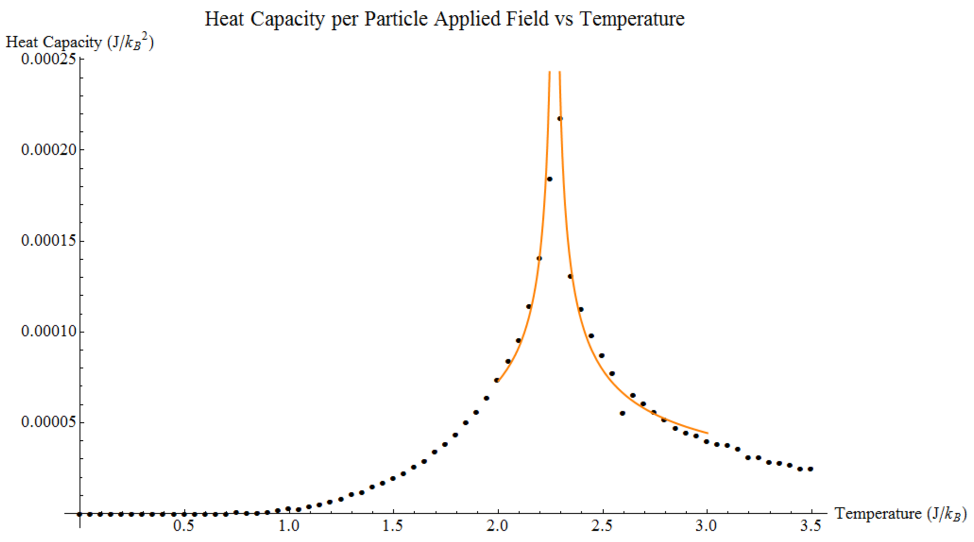
Repeating this process for susceptibility and fitting the data for the last 6 points on the right hand side (orange) of Tc resulted in the following plot:

Below: Susceptibility vs Temperature with right hand side fit in orange for N = 10,000 and temperature step size of 0.05



We can see the fit on the right side is quite good, and is in agreement with theory. Fitting τ and κ, to the function $\chi = \kappa\,|T-\tau|^{-\frac{7}{4}}$, gives the following fit: $\frac{\mathbf{1.71\times10^{-4}}}{|\mathbf{2.262}-\boldsymbol{T}|^{7/4}}$. The fit give a value for the critical temperature, **τ = 2.262** is within 1% of the theoretical value the fit overall has a good agreement with the critical exponent. The asymmetry of the plot and stretching of the right hand side is most likely due to the residual magnetisation. Ignoring this affect may have resulted in a more symmetrical fit, but this would only affect κ and not the value of the critical exponent.

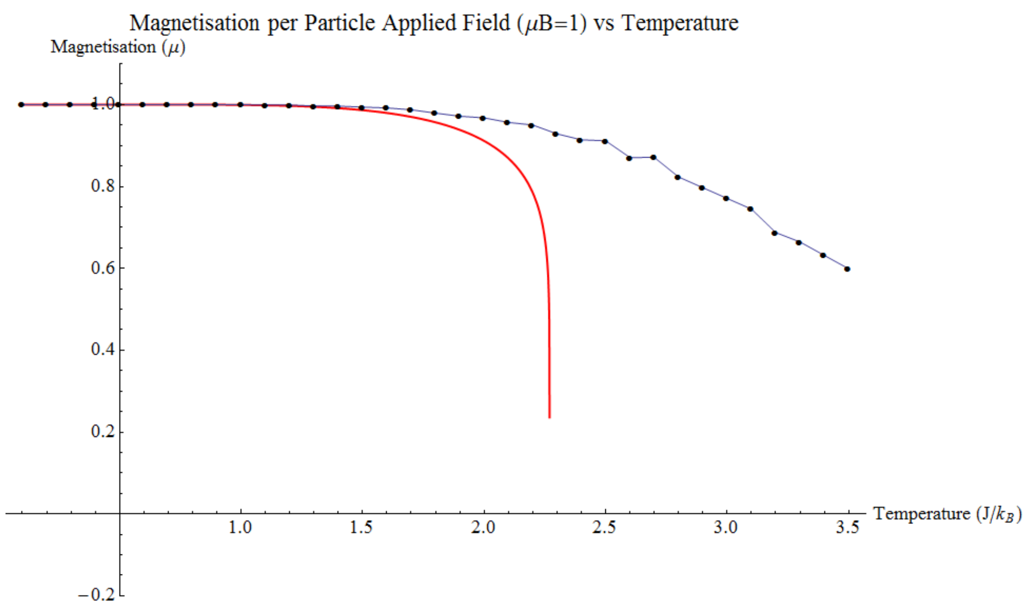
Finally, I could not fit the heat capacity for a value of α = 0, which suggests heat capacity should be constant. This is clearly not the case. However, another source [9, p. 22], suggests alpha should be interpreted differently for a 2D Ising model and effectively be $\alpha = -0.5$. This relationship fits well as shown below (orange) with a constant of proportionality, **κ = 2.8 × 10^{-5}**. Repeating and fitting only the right side for $C = \kappa\,|T-\tau|^{-\frac{1}{2}}$ we find that the critical temperature, **τ = 2.282 and κ = 3.6 × 10^{-5}.** This fitted value for the critical temperature is also within 1% of the theoretical value indicating a good fit.



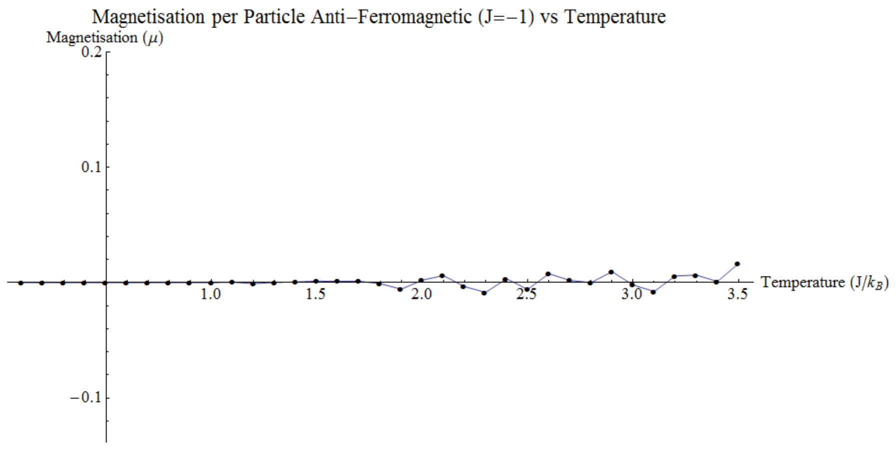
More advanced methods for finding the critical exponents exist, such as 'finite-size scaling' which is based on finding values for macroscopic properties at different lattice sizes and linear fitting these results to find values for the critical exponents. I could not get this method to reliably work in the time I had.

Interesting Extensions – Applied external field and Anti-ferromagnetism:

If we apply an external magnetic field by setting the value of μB to 1 and recompiling and running with default settings the model produces the graph below for magnetisation (zero-field, theoretical phase transition shown in red). We can see that there is no sudden phase transition below the critical temperature; instead the magnetisation slowly declines with temperature. This is expected, as the material becomes paramagnetic above the critical temperature and the spins align with the external magnetic field. As temperature increases thermal excitation randomly orientates the spins and gradually reduces overall magnetisation. One would expect that with greater temperature the magnetisation will asymptotically approach 0.



Setting the exchange energy parameter, J, to -1 result in anti-ferromagnetic behaviour [3], meaning neighbouring spins orientate themselves in opposing directions. We would expect a zero magnetisation at all temperatures. As we can see from the graph below, the simulation reproduces these results with zero magnetisation for low temperatures, and very small deviations from zero at higher temperatures due to thermal noise causing small, random magnetisations.



Conclusion:

The simulation has proven to be accurate and efficient, with results produced having a good agreement with theoretical models and other implementations of 2D Ising model simulators. Results were shown to match the predicted critical exponents and a distinct phase transition at the theoretical temperature is apparent. The implementation code is modular, relatively efficient and is designed to be easily expandable for future improvements. Finally, the implementation was shown to be sufficiently general, handling cases for applied magnetic fields and anti-ferromagnetism.

Works Cited

[1] H. Gould, J. Tobochnik and W. Christian, “Monte Carlo Simulation of the Canonical Ensemble,” in *An Introduction to Computer Simulation Methods: Applications to Physical Systems (Draft)*, 2002, p. Chapter 17.

[2] Wikipedia, “Algorithms for calculating variance,” 12 January 2015. [Online]. Available: http://en.wikipedia.org/wiki/Algorithms_for_calculating_variance. [Accessed 18 January 2015].

[3] Wikipedia, “The Ising Model,” January 16 2015. [Online]. Available: http://en.wikipedia.org/wiki/Ising_model. [Accessed 18 January 2015].

[4] I. Agarwal, “Numerical Analysis of 2-D Ising Model,” 2011 March 2011. [Online]. Available: http://www.hiskp.uni-bonn.de/uploads/media/ising_II.pdf. [Accessed 18 January 2015].

[5] L. Witthauer and M. Dieterle, “The Phase Transition of the 2D-Ising Model,” 2007. [Online]. Available: http://quantumtheory.physik.unibas.ch/people/bruder/Semesterprojekte2007/p1/Ising.pdf. [Accessed 18 January 2015].

[6] N. Drakos, R. Moore and R. Fitzpatrick, “The Ising Model,” The University of Texas at Austin, 30 March 2006. [Online]. Available: http://farside.ph.utexas.edu/teaching/329/lectures/node110.html#image5. [Accessed 18 January 2015].

[7] W. Janke, “Monte Carlo Simulations of Spin Systems,” Institut f ̇ur Physik, Johannes Gutenberg-Universit ̈at, D-55099 Mai, 28 April 2011. [Online]. Available: http://www.physik.uni-leipzig.de/~janke/Paper/spinmc.pdf. [Accessed 18 January 2015].

[8] M. Medo and Y.-C. Zhang, “Simulations for the Ising model (and more), Numerical Methods for Physicists, Lecture 6,” 25 March 2013. [Online]. Available: http://tinyurl.com/ozkevha. [Accessed 21 January 2015].

[9] J. Kotze, “Introduction to Monte Carlo methods for an Ising Model of a Ferromagnet,” 3 March 2008. [Online]. Available: http://arxiv.org/pdf/0803.0217.pdf. [Accessed 21 January 2015].