Simulating the 2D Ising Model and Extensions

Newton Cheng and Kaitlyn Shin
Department of Physics, Stanford University, Stanford, California 94305, USA
(Dated: May 31, 2018)

In this paper, we review the basics of Markov chain Monte Carlo techniques, and then proceed to apply said techniques to simulate the 2D Ising model using the Metropolis and Wolff cluster algorithms. Using our simulation, we find evidence of a phase transition and critical behavior at a critical temperature of $T_c = 234123412$ and values of the critical exponents $\alpha = 2141$, $\beta = 234532$, and $\gamma = 2345234$. These results agree with the analytic values to 123421%. We then proceed to extend our work by considering other models with similar techniques, such as the anti-ferromagnetic 2D Ising model, the $q = 3$ Potts model, and the XY model. In all cases, we find evidence of critical behavior, and are able to extract information that agrees to good precision with analytic studies of the behavior. We conclude by considering possible improvements to our simulations, as well as possible further avenues of exploration.

INTRODUCTION

Note that we use natural units with $k_B = 1$.

2D ISING MODEL

The 2D Ising model on a lattice is defined by the Hamiltonian:

$$H = -J \sum_{\langle ij \rangle} s_i s_j - h \sum_i s_i \tag{1}$$

where s_i is the ith spin on the lattice, $\langle ij \rangle$ refers to nearest neighbors on the lattice, J is the coupling strength between the spins, and h is a coupling to an external magnetic field. Without loss of generality, we can take $s_i = \pm 1$ (just rescale J), and we will be concerned with the basic case of the square lattice. Notice that the energy is determined only by local properties of the total spin configuration, and the sign of J determines if spins preferentially align or anti-align. When $J > 0$, the model is ferromagnetic, and $J < 0$ corresponds to an anti-ferromagnetic model. We will only be concerned with the case of no external magnetic field, so we set $h = 0$.

The partition function is

$$Z = \sum_{\{s_i\}} e^{-\beta H} \tag{2}$$

where $\{\sigma_i\}$ is the set of all 2^N spin configurations of the lattice with N sites. Some quantities we will be interested in are the mean energy and the mean magnetization per spin:

$$\langle E \rangle = -J \left\langle \sum_{\langle ij \rangle} s_i s_j \right\rangle \tag{3}$$

$$\langle m \rangle = \frac{1}{N} \langle M \rangle = \frac{1}{N} \left\langle \sum_i s_i \right\rangle \tag{4}$$

We will also be interested in the specific heat per spin and the magnetic susceptibility:

$$c = \frac{1}{N} \frac{\partial \langle E \rangle}{\partial T} = \frac{\beta^2}{N} (\langle E^2 \rangle - \langle E \rangle^2) \tag{5}$$

$$\chi = \frac{1}{N} \frac{\partial \langle M \rangle}{\partial B} = \frac{\beta}{N} (\langle M^2 \rangle - \langle M \rangle^2) \tag{6}$$

We can understand some of the properties of the model by examining the high and low temperature limits of the model with N lattice sites. At low temperatures, the two ground states of fully aligned spins dominate the partition function. The energy per spin of an infinite lattice or lattice with periodic boundary conditions with coordination number z is simply $E/N = -Jz/2$, so for our square lattice with $z = 4$, we have $E/N = -2J$. We also expect that the mean magnetization per spin should be 1 or -1, as is the case for the two ground states.

At high temperatures, every spin configuration is equally likely, so the highest entropy configurations dominate. We therefore expect that there are approximately equal numbers of up and down spins, the mean energy per spin is 0, and that the magnetization $\langle s \rangle$ is 0. This can be analytically confirmed by performing a high temperature expansion of the partition function, upon which one finds that only closed paths on the lattice (in terms of active bonds) contribute.

One of the primary motivations for studying the 2D Ising model is the presence of a phase transition; this is evident from the change of the order parameter $\langle m \rangle$ at high and low temperatures. The critical temperature can be analytically computed using the self-dual property of the 2D Ising model on the square lattice; every term in the partition function's high temperature expansion can be mapped in a unique fashion to the partition function's low temperature expansion. The temperature at the "crossing point" can be computed to be $T_c/J = \frac{2}{\ln(\sqrt{2}+1)}$. As a check on our simulation later, we will compute various thermodynamic quantities of the system on both sides of the critical temperature, and check that we obtain the expected results.

We will also be interested in analyzing the critical behavior itself. The phase transition is second-order, with divergences in the the specific heat, magnetization per spin, and magnetic susceptibility. The associated critical exponents can be analytically computed as $\alpha = 0$, $\beta = \frac{1}{8}$, and $\gamma = \frac{7}{4}$, respectively. Ideally, our simulation will be able to reproduce all 3 critical exponents.

MARKOV CHAIN MONTE CARLO

In thermal systems, we are generally interested in the expectation value of some observable Q, rather than the precise state of the system. Generically, the expectation value can be computed with the system's partition function:

$$\langle Q \rangle = \frac{\sum_n Q_n e^{-\beta E_n}}{\sum_n e^{-\beta E_n}} \tag{7}$$

Beyond the absolute simplest models, analytically computing such a quantity is intractable. The idea behind using Monte Carlo techniques to simulate such systems is that one samples states according to some specified probability distribution, and then obtains an estimate of $\langle Q \rangle$ from the sample:

$$Q_M = \frac{\sum_{n=1}^{M} Q_n \frac{e^{-\beta E_n}}{p_n}}{\sum_{n=1}^{M} \frac{e^{-\beta E_n}}{p_n}} \tag{8}$$

One possible choice is to sample all possible states with equal probability, so $p_n = 1$ for all n. This is generally a poor choice, because outside of extremely high temperatures, there are relatively few states that dominate the partition function at any given temperature due to the exponential suppression of the Boltzmann factor, so most of the chosen sampled states in the estimator above will contribute nothing to the sum. Correspondingly, the estimate itself will be highly inaccurate unless one samples a large number of states.

A clever choice of probability distribution is the Boltzmann distribution itself $p_n = e^{-\beta E_n}/Z$, so that the dominant states are sampled most frequently, and we obtain a more accurate estimate while looking at fewer states. The estimator becomes

$$Q_M = \frac{1}{M} \sum_{n=1}^{M} Q_n \tag{9}$$

which is much more tractable. This general strategy is called importance sampling. The goal is then to develop a systematic method of selecting states according to the Boltzmann distribution.

The obvious choice is to randomly choose states, which we then accept with probability $e^{-\beta E_n}$. This runs into the same problem as before, where almost all states have a minuscule probability of acceptance. A better choice is to use a Markov process, which generically refers to a process by which a given state X transitions to a state Y with transition probability $P(X \to Y)$. The transition probabilities for a Markov process are constant, depend only on the initial state X and candidate state Y, and sum to 1. Repeatedly applying such a process to evolve a system generates a Markov chain of states. The next step is to select the transition probabilities to reflect the Boltzmann distribution.

The fact that we hope to simulate a thermal system creates two constraints on the transition probabilities: ergodicity and detailed balance. Ergodicity is the requirement that any state be obtainable from any other state using the algorithm. Detailed balance is the requirement that the rate a system transitions from a state X to a state Y is equal to the rate that a system will transition from Y to X, which we model as

$$p_X P(X \to Y) = p_Y P(Y \to X) \tag{10}$$

We want our sample probabilities p_n to satisfy the Boltzmann distribution, so we constrain the transition probabilities to satisfy:

$$\frac{P(X \to Y)}{P(Y \to X)} = e^{-\beta(E_Y - E_X)} \tag{11}$$

The total transition probability of moving from state X to state Y is the product of these two probabilities: $P(X \to Y) = g(X \to Y)A(X \to Y)$. The selection probability g refers to the probability distribution from which we select a new state, and the acceptance ratio A refers to the probability that we accept the new state and transition to it from our current state. An algorithm refers to a particular choice of the selection probability distribution and the acceptance ratios.

METROPOLIS ALGORITHM

Theory

The first algorithm we will use to simulate the 2D Ising model is perhaps the most historically important: the Metropolis algorithm, introduced by Nicolas Metropolis and his colleagues in 1953. This is also the most well-known example of a single-spin-flip algorithm, so-called because the state-to-state evolution occurs by flipping one spin each time. The selection probabilities of all states that differ from the current state by one spin are equal, and the selection probability of all other states are 0. More concretely, consider a lattice of N spins in a configuration we call X. There are a set of spin configurations $\{S_i\}$ that differ from X by one spin. Then the

selection probability is

$$g(X \to Y) = \begin{cases} \frac{1}{N}, & Y = S_i \\ 0, & \text{else} \end{cases} \tag{12}$$

Detailed balance then implies that the acceptance ratios must satisfy:

$$\frac{P(X \to Y)}{P(Y \to X)} = \frac{A(X \to Y)}{A(Y \to X)} = e^{-\beta(E_Y - E_X)} \tag{13}$$

The simplest choice would be to simply set $A(X \to Y) = A_0 e^{-\frac{1}{2}\beta(E_Y - E_X)}$ for some constant A_0, which turns out to be a maximum at $e^{-4\beta J}$ for the square lattice. Unfortunately, this is extremely slow and inefficient, because almost all acceptance probabilities are 0; in particular, when $E_Y - E_X > 0$, there is almost no chance of acceptance.

The Metropolis algorithm solves this issue by offering an acceptance ratio:

$$A(X \to Y) = \begin{cases} e^{-\beta(E_Y - E_X)}, & E_Y - E_X > 0 \\ 1, & \text{else} \end{cases} \tag{14}$$

In other words, if the selected configuration has a lower energy than the current one, it is automatically accepted. Otherwise, it is accepted with a probability determined by the Boltzmann factor. This acceptance ratio manifestly obeys the detailed balance constraint, and has the advantage of being significant faster, because it spends far less time rejecting states than the simplest choice. Ergodicity is also clear with this algorithm, as it is clearly possible to reach any state from any other by systematically flipping individual spins.

Implementation

Our simulation is performed entirely in Python. For all simulations, we use units such that $J = 1$.

Taking our spins to have value ± 1, we define a 2D array with elements that can take on ± 1 to represent our array. We take the array to have periodic boundary conditions, so that we can neglect edge effects. The starting state for almost all of our simulations is the $T = 0$ state with all spins aligned. The other option is to use a $T = \infty$ state, where every spin has a 50/50 probability of taking on either $+1$ or -1.

Now we apply the Metropolis algorithm. To do this, we use a program that performs the following steps:

1. Randomly select a spin on the given lattice
2. Flip the spin and compute the change in energy of the lattice ΔE
3. If $\Delta E < 0$, accept the new state
4. If $\Delta E > 0$, generate a random number X from a uniform distribution $[0, 1)$
5. If $X < e^{-\beta \Delta E}$, accept the new state; otherwise reject the new state

We may then iterate over this function n times to perform n timesteps of evolution. It is useful to introduce the time-unit *sweep*, corresponding to N timesteps, where N is the number of lattice sites. The attempt frequency of flipping any particular spin is then independent of the lattice size, and we will use this as our preferred unit of time. Some pictures of the evolution of a lattice are shown in Fig. 1.

Optimizations

Because we will be interested in performing long running simulations across multiple temperatures, we would like to make the simulation as efficient as possible. Towards that end, we implement several optimizations.

First, we take advantage of the fact that the energy of a spin on the lattice is a local property of the lattice. Computing the energy of the entire lattice is generically a time-consuming process; the number of bonds on the lattice is $N_B = 2N$, so the energy will be given by a sum with that many terms. Once the number of iterations becomes large, this will be a highly expensive sum to compute at every step. Instead, we can directly compute the change in energy, independent of the total energy. The difference in energy between two states X and Y that differ by one spin s_k is

$$E_Y - E_X = -J \sum_{\langle ij \rangle} s_i^Y s_j^Y + J \sum_{\langle ij \rangle} s_i^X s_j^X \tag{15}$$

$$= -J \sum_{\langle ik \rangle} s_i^X (s_k^Y - s_k^X) \tag{16}$$

The change in energy is just a sum over the nearest neighbors to s_k. Then using the fact that $s_k^Y = -s_k^X$, we conclude

$$\Delta E = 2J s_k^X \sum_{\langle ik \rangle} s_i^X \tag{17}$$

This is just a sum over 4 terms and depends only on the state prior to flipping the spin. Evidently this is far more efficient than summing over the entire lattice at every step.

Given that we compute the change in energy at every step, we also have a way to efficiently measure the energy at every timestep: compute the energy of the initial lattice E_i, then just increment that number by the change in energy with every timestep $E_j = E_i + \Delta E$. This is far more efficient than computing the energy of the entire lattice at every step.

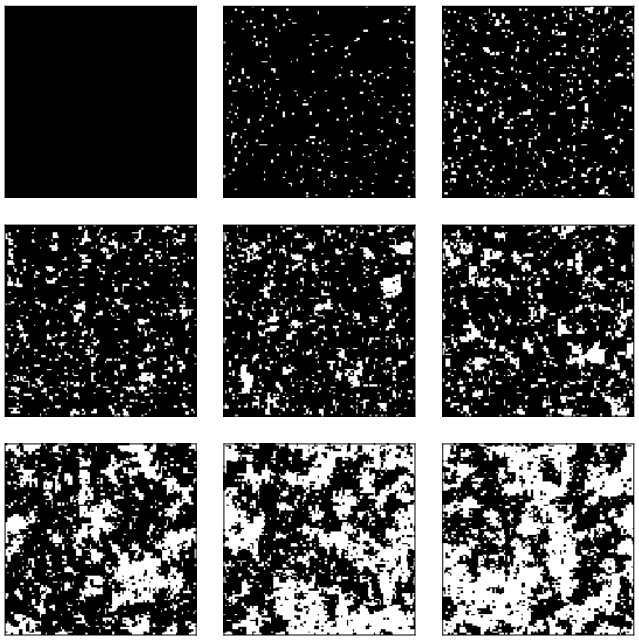

FIG. 1. The evolution of a 100×100 square lattice at $T = 2.4$ via the Metropolis algorithm. The sequence is ordered from left to right, top to bottom. The two spin states $+1$ and -1 correspond to black and white, respectively. We have initialized the system in the $T = 0$ state, and plotted the system after 0, 1, 5, 10, 20, 50, 100, 500, and 1000 Monte Carlo sweeps.

We may use a similar trick for keeping track of the magnetization of the lattice. Generically, we need to sum over the entire lattice, but we know that the change in magnetization of flipping, say, spin s_k is simply $\Delta M = 2s_k$. Then we just compute the magnetization of the lattice at the beginning, and increment that value by ΔM every time a spin is flipped.

Because the change in energy is limited to 4 terms that are $\pm 2J$ (the change per bond), the entire spectrum of energy changes is simply $\{-8J, -4J, 0, 4J, 8J\}$. Then instead of computing the Boltzmann factor $e^{-\beta \Delta E}$ at every iteration, which is very computationally expensive due to the exponential, we can compute the factors once at the beginning of the simulation for a given temperature. Moreover, we do not even need to compute the Boltzmann factor for all the energy changes, just the ones that are positive. We then need only compute 2 exponentials, instead of $5n$, where n is the number of iterations.

Equilibration and Measurement

Given that our initial lattice is at $T = 0$ or $T = \infty$, we have to allow the simulation to equilibrate at our desired temperature T before taking any measurements. For example, it's clear that the $T = 0$ configuration is not an equilibrium configuration for $T = 100$, because several spins will flip within the first several iterations. We may also refer to Fig. 1: any measurements made in the first 6 states are not accurate with respect to the final (most accurate) state. For our simulation, we take equilibration this to mean that thermodynamic quantities whose mean values depend on the system temperature are relatively stable within a narrow range. This is purposefully a vague definition; we are ultimately more interested in the thermodynamic quantities of the state, not the specific state itself, and we expect thermal fluctuations about the mean. In our simulation, we consider 2 quantities: the energy per spin and magnetization per spin. An example of how these quantities will evolve over time is shown in Fig. 2.

We see that for the first part of the simulation, the energy and magnetization rapidly change; clearly any measurements taken at any point in that time range will be inaccurate. However, after roughly 500 sweeps, we see that the energy and magnetization become stable, oscillating around a mean value. Such behavior signals equilibration. In all our simulations, we first perform such a check to ensure that our system has equilibrated before beginning to take measurements. Moreover, because we are interested in a small range of temperatures, one can equilibrate beginning from the final state of the previous temperature. Because the states will presumably be fairly close, equilibration occurs faster.

We want to obtain mean values for various quantities like the energy. The natural way to do this is to take a series of measurements at the same temperature, and then find the arithmetic mean of the measurements. However, such a process only yields the true mean if each measurement is independent of the previous one. Because

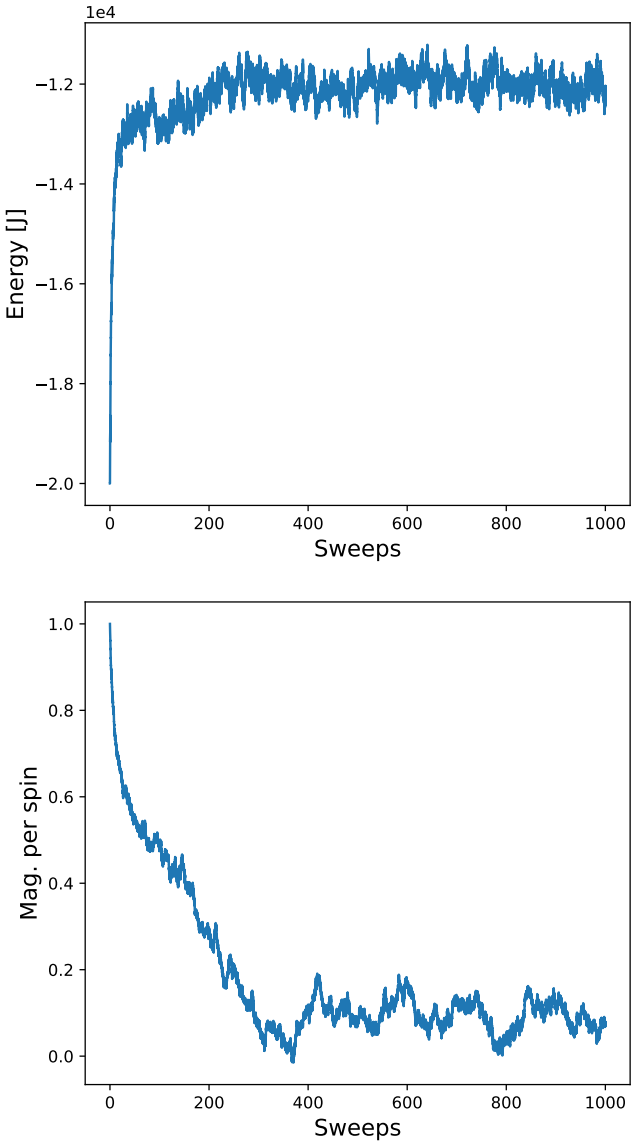

FIG. 2. Plots of the total energy and the magnetization per spin on a 100×100 square lattice at $T = 2.4$ initialized at $T = 0$ and evolved with the Metropolis algorithm. We see that after roughly 500 sweeps, the system reaches equilibrium, with the energy hovering around $-1.2 \cdot 10^4$ and the magnetization per spin around 0.1.

our simulation utilizes single-flip dynamics, it seems clear that measurements taken too soon together will be correlated. In an extreme case, one can imagine measuring the magnetization of the system after just one Monte Carlo step, so that the second measurement is either the same as the first one or differs by ± 2, which is a very strong correlation. So ideally we need to wait long enough between measurements that the state we measure from is significantly different from the first.

To obtain the correlation time, we can, for example, compute the magnetization auto-correlation defined by

$$\chi(t) = \int dt' [m(t')m(t'+t) - \langle m \rangle^2] \tag{18}$$

To get statistically independent measurements, we would ideally like to wait twice the correlation time 2τ, where τ is the time needed for the auto-correlation to drop by a factor of $\frac{1}{e}$. Unfortunately, we immediately run into the issue of time: τ is at least as large as 1 sweep (formally giving every spin the chance to flip), but usually larger. For practical purposes, we take measurements after every sweep; although our measurements are not entirely uncorrelated, we will be more interested in behavior across a temperature range than at a specific temperature, so the trend of the data will be more important than their specific values. We also rely on the fact that expect our data to fall within a relatively narrow range of values to begin with (this naturally becomes less accurate as fluctuations grow).

While the Metropolis algorithm is extremely efficient at very high and very low temperatures, the critical temperature region poses a major difficulty for two reasons: critical fluctuations and critical slowing down. As one approaches the critical temperature, the presence of spin clusters can cause significant statistical error and large fluctuations in the region of interest. One can somewhat mitigate this issue by approaching the critical temperature from below, but statistical errors will still tend to be large around the critical temperature. Formally, one deals with statistical errors by simply averaging over a larger sample size. However, as one approaches the critical temperature, the correlation time will generally scale as $\tau \propto \xi^z$, where ξ is the correlation length and z is the critical exponent and has been measured to be approximately 2.17 for the Metropolis algorithm. At T_c, the correlation length is the size of the lattice itself (so diverges in the thermodynamic limit), implying that large numbers of measurements will take an impractical amount of time to obtain. Critical slowing down is an innate aspect of the Metropolis algorithm, so there is nothing much that can be done to deal with the issue. We will see that the Wolff algorithm is much more efficient around the critical temperature, because critical slowing down is much less pronounced.

Error

The error in our measurements is computed using the bootstrap method – the benefit of this method is that it does not require knowing the correlation time, nor does it require truly independent measurements. Moreover, it also readily applies to direct measurements, like mean energy, and to functions of means, like the specific heat.

Using the specific heat as an example, the idea is to resample our total sample of measurements and compute

the specific heat from that resample, then repeat the process several times. As an example, suppose we are given 100 measurements of the energy that we would like to use to compute the specific heat. Then we choose 100 values from the measurements with replacement, so that we may pick up duplicates, and compute the specific heat from those 100 values. We then repeat the process, and after several iterations, say 1000, the standard deviation of the bootstrap values approaches the true error:

$$\sigma^2 = \bar{c^2} - \bar{c}^2 \tag{19}$$

Results and Accuracy Check

As a first check, we want to make sure that the system behaves as we would expect at low and high temperature limits. In the low temperature limit, all the spins should be aligned so that the energy is at the minimum $E_0 = -2N$ and the magnitude of the magnetization is 1. In the high temperature limit, spins are randomly aligned so that both the energy and the magnetization are 0. Some examples on a 100×100 grid are shown in Figure **FIGURE**. We see that we have the expected behavior across the critical temperature, with a progression of total randomness to domain formation to full alignment going from high to low temperature.

A stronger check of the accuracy of simulation is how well it can reproduce the expected critical behavior of the 2D Ising model. To do this, we explore a range of temperatures, $T = 1.6$ to $T = 2.9$, around the known critical temperature $T_c = 2.27$ in .01 increments, as seen in Figure **FIGURE**. At every temperature value, we allow the system to equilibrate over $100N$ sweeps for a $N \times N$ lattice, and then proceed to make $50N$ measurements of the energy and magnetization, with each measurement separated by one sweep. The measurements can then be averaged to obtain the specific heat and magnetic susceptibility. All the plots of these values for every lattice we explored can be found in the **APPENDIX**.

There are several caveats to keep in mind about our results. The first is that the finite temperature resolution size ultimately limits the precision of our results, so we cannot really trust our results beyond 3 significant figures. The equilibration time value is somewhat arbitrarily chosen using the heuristic that it takes order N^3 steps to equilibrate, and is probably not enough for temperatures very close to the critical temperature. However, we would like to explore intermediate-sized lattices, so we opt to trade accuracy in the computation for accuracy in the system size. From looking at the various plots for different lattice sizes, increasing the lattice size also greatly improves the precision of our results and reduces the error on each data point. The instability of the simulation near the critical temperature is particularly pronounced for the smaller lattices because the critical fluctuations affect a larger fraction of the lattice, and extracting information about the critical behavior would be close to impossible.

Using our values for the magnetization per spin and magnetic susceptibility, we perform a non-linear least squares fit to a function of the form (coming from below the critical temperature)

$$f(x) = a\left(\frac{b-x}{b}\right)^c \tag{20}$$

using the SciPy `optimize.curve_fit` function. The fit parameters b and c correspond to the critical temperature and critical exponent, respectively. We can also fit the specific heat per spin to this form, but the analytic solution implies a logarithmic divergence. Naturally the finite size of our lattice will affect the values we obtain for the critical exponents, as we will not see true divergences. Furthermore, in the case of the specific heat, we are also not guaranteed that the maximum we see is actually at the critical temperature. It is also for this reason that we are looking to explore the largest lattices we can that do not require excessive computational time.

The largest lattice we consider is a 50×50 lattice, and we find critical values:

WOLFF ALGORITHM

ANTI-FERROMAGNETIC ISING MODEL IN 2D

By setting $J < 0$ in the Ising Hamiltonian in equation (1), we obtain the anti-ferromagnetic Ising model. Aligned spins then have a positive energy contribution, so spins will preferentially anti-align. The two ground states of the system then correspond to the two "checkerboard" patterns on the lattice, where every spin is surrounded by the opposite spin. The order parameter is now the *staggered magnetization*, whose value on a spin at the i, j location on the lattice is

$$m^\dagger_{ij} = (-1)^{i+j} s_{ij} \tag{21}$$

Essentially, we assign a sign to each lattice site that multiplies against the spin. For our simulation, we assign $+1$ to the spins whose indices on the 2D lattice sum to an even number. The magnitude of the total staggered magnetization is then maximized precisely in the checkerboard ground state, and averages to 0 for high temperatures.

We can easily adapt the code we use for the ferromagnetic Ising model for the anti-ferromagnetic model – simply change the sign of J. The results of the simulation are shown in Figure **FIGURE**. We find evidence of a second-order phase transition at a critical temperature of $T_c =$ **A BILLION** and critical exponents **A BILLION**.

POTTS MODEL

Theory

A natural generalization of the Ising model is to consider spins with more possible states. Taking q to be a positive integer, the q-state Potts model is defined by the Hamiltonian

$$H = -J\sum_{\langle ij\rangle}\delta_{s_i,s_j} - h\sum_i s_i \tag{22}$$

where δ_{ij} is the Kronecker delta which takes the value 1 when $i = j$ and is 0 otherwise. As before, we will only consider the case of a vanishing external field and set $h = 0$. The Hamiltonian is again the sum of the energy in bonds between spins, with bonds between identical spins contributing $-J$ and bonds between non-identical spins contributing 0. By rewriting the Hamiltonian, we can see that the Ising model is equivalent to the $q = 2$ Potts model up to an energy rescaling and an additive constant:

$$H = -\frac{1}{2}J\sum_{\langle ij\rangle}2\left(\delta_{s_i,s_j} - \frac{1}{2}\right) - \frac{1}{2}\sum_{\langle ij\rangle}J \tag{23}$$

On comparison with the Ising model, we find a couple differences: there are now q-many ground states, corresponding to all spins on the lattice taking on the same value. Furthermore, the entropy of a Potts model with $q > 2$ is significantly higher at all temperatures, because there are far more possible microstates.

The order parameter in the Potts model is defined by:

$$m = \frac{1}{N}\left|\sum_{k=1}^{q} e^{2\pi ik/q}n_k\right| \tag{24}$$

where n_k is the number of spins in state k. The magnetization of a particular spin is given by a generically complex q-th root of unity, and the total magnetization is then the magnitude of the sum of all the magnetizations. In the ordered phase, spins will preferentially align so that $n_k \to N$ for some particular $k \in [1, q]$, $n_j \to 0$ for all $j \neq k$, and $m \to 1$. In the disordered phase, we will have $m \to 0$ due to phase cancellation. Therefore m is 0 below the critical temperature and is 1 above it, as desired. Note that in the special case of $q = 2$, the roots of unity are ± 1, so we recover the order parameter used in the Ising model.

Implementation

The Metropolis algorithm may still be applied to simulate the q-state Potts model: randomly select a spin s_i and a value for the spin $s_i' \neq s_i$. Then compute the change in energy ΔE resulting from the change $s_i \to s_i'$. If $\Delta E < 0$, we accept the new state. If $\Delta E > 0$, we accept the new state with probability $e^{-\beta\Delta E}$. All of our previous optimization comments remain true: one need only compute the energy of the full lattice at the beginning of the simulation and increment by ΔE at each step, and the spectrum of possible ΔE is small enough that computing the Boltzmann factors at the beginning of the simulation is highly efficient.

While the Metropolis algorithm is reasonably efficient for low q, it significantly slows down as q increases. For example, for $q = 100$ and low temperatures, consider a spin s_i whose neighbors all have different values. There are at most 4 values of s_i' that will decrease the energy of the lattice, so there is at most a 4/99 chance that the algorithm will select a spin value that lowers the energy. Because the other 95/99 choices for s_i' have $\Delta E = 0$, they will always be accepted. The algorithm will spend most of the time accepting new configurations that do not give rise to any dynamics, because the ratio of interesting states to possible states is so high. On the other hand, suppose s_i does match one of its neighbors. Then there are at 96/99 choices for s_i' that yield $\Delta E > 0$, and at low temperatures, the acceptance ratio is very low. The algorithm will spend most of its time rejecting these higher energy states.

A more efficient algorithm for higher q is the heat-bath algorithm, which assigns chooses states as if in a heat bath at the given temperature. More precisely, the acceptance ratio is 1 for every state, while the selection probability for each state $n \in \{1, \ldots, q\}$ is

$$p_n = \frac{e^{-\beta E_n}}{\sum_{i=1}^{q} e^{-\beta E_i}} \tag{25}$$

This manifestly satisfies detailed balance, and will preferentially select states that evolve the system, rather than spend a significant amount of time sorting through un-dynamical states. The algorithm can then be implemented in the following steps:

1. Random select a spin on a given lattice
2. Choose a new value n for the spin based on the probability distribution p_n defined above
3. Return the lattice with the new value for the spin

The selection probability distribution also has the advantage of being relatively easy and efficient to implement. The Hamiltonian can be split into a two parts: one that depends on the spin in question s_k, and one that does not.

$$H = -J\sum_{\langle ij\rangle}\sum_{i,j\neq k}\delta_{s_i,s_j} - J\sum_{\langle ik\rangle}\delta_{s_i,s_k} \tag{26}$$

The first term will be the same in all the exponentials in p_n, so it cancels out. We therefore need only sum

over 4 terms for each exponential. Moreover, we do not actually need to compute the energy in every case; clearly there can only be at most 4 values of s_k that will yield a non-zero contribution to the Hamiltonian. Therefore, there are only 4 exponentials in p_n that actually need to be computed, with the rest just having value 1. In computing the energy for those 4 exponentials, there are only 4 non-zero values for ΔE: $\{-J, -2J, -3J, -4J\}$. Because the spectrum is so small, we can use the same trick as in our other simulations, where we compute the exponentials once at the beginning of the simulation and just pull the values as needed.

A comparison of the equilibration times for the Metropolis and heat-bath algorithms for the $q = 10$ model is shown in Figure **FIGURE**. Although one step of the heat-bath algorithm is slower than the Metropolis algorithm by a $\mathcal{O}(1)$ factor, the equilibration time is faster by a factor of **A BILLION**, representing a significant gain in efficiency.

It is worth noting that for small q, when there are only a few un-dynamical states, the Metropolis algorithm may be faster. As an example, we show equilibration times for a $q = 3$ model in Figure **FIGURE**. The Metropolis algorithm not only equilibrates faster, but each steps is also faster than the heat-bath algorithm, so it is clearly faster overall.

Results

XY MODEL

Whereas the Ising model represents a simple system of spins constrained to 1 dimension ($n = 1$), the XY model represents a system of with interacting two-dimensional vector spins ($n = 2$). The 2D XY model on a lattice is defined by the Hamiltonian:

$$H = -J \sum_{\langle ij \rangle} \cos(\theta_i - \theta_j) - h \sum_i \theta_i \tag{27}$$

where θ_i is the ith spin on the lattice, $\langle ij \rangle$ refers to nearest neighbors on the lattice, J is the coupling strength between the spins, and h is a coupling to an external magnetic field **CITE**. The state θ_i can belong to the region $[0, 2\pi)$. Again, without loss of generality, we work with the basic case of the square lattice with the case of no external magnetic field ($h = 0$). When $J > 0$, the model is ferromagnetic, and when $J < 0$, the model is anti-ferromagnetic.

This system is of great theoretical interest because of its circular symmetry. As a result of the Mermin–Wagner Theorem, in this $n \leq 2$ system, continuous symmetries cannot be spontaneously broken at finite temperatures with short-range interactions within the system **CITE**. However, although there is no second-order phase transition (the derivative of the specific heat per spin c_v is continuous), there is a transition from a high-temperature disordered phase to a low-temperature quasi-ordered phase. In the former disordered phase, the correlation function decays exponentially; in the latter quasi-ordered phase, most of the spins tend to be aligned and the correlation function has a power law decay—the exponential correlation length is infinite. This transition between the two phases, called the *Kosterlitz–Thouless transition* (hereafter the KT transition), occurs at some critical temperature T_c and is of infinite order **CITE**.

The degrees of freedom determining how ordered the system is come from topologically stable configurations in the 2D XY model called *vortices*, which can be visualized as local sources/sinks (vortices) and saddles (antivortices) of multiple spins **FIGURES**. In other words, these vortices are points on the lattice around which the spins are configured in a winding pattern, clockwise or counterclockwise. Below T_c, there are only a few vortices, which present themselves in bound vortex-antivortex pairs of net zero vorticity **FIGURE**. Above T_c, there are many free vortices present in what can be called a vortex plasma. Vortex generation has been found to be the reason behind the exponential correlation decay at the high-temperature disordered phase **CITE**. Since these vortices are not bound, they can move due to the influence of an applied magnetic field.

Thus, the KT transition occurs because of vortex-antivortex pairs unbinding at T_c, which causes a sudden change in the response form to an arbitrarily weak applied magnetic field. This transition is observable in simulations of the 2D XY model, though large lattice sizes and long simulation times are often necessary to simulate and observe the transition accurately. The 2D XY model can be used to model systems that possess similar properties of symmetry and exhibit similar transition behavior, such as superfluidity in thin films—observed experimentally with He^4 **CITE**.

Implementation

The Metropolis algorithm may again be used to simulate the 2D XY model. For all simulations, the starting states consist of randomly arranged spins on a lattice with periodic boundary conditions, with the option of seeding available for result reproducibility. As with the Metropolis algorithm implementation for the 2D Ising model simulation, we implement a program in Python that follows the following steps:

1. Randomly select a spin on the given lattice
2. Change the spin to a random orientation and compute the change in Hamiltonian energy of the lattice ΔH
3. If $\Delta H < 0$, accept the new state

4. If $\Delta H > 0$, generate a random number X from a uniform distribution $[0, 1)$

5. If $X < e^{-\beta \Delta H}$, accept the new state; otherwise reject the new state

As before, the energy of a spin on the lattice is a local property with four neighboring spins affected. This algorithm is iterated for N units of time, with n steps of evolution per sweep.

A Wolff clustering algorithm was considered, but due to time constraints was unable to be implemented. However, such an algorithm for the 2D XY model would have been implemented similarly to the Wolff clustering algorithm implemented for the 2D Ising model. A random spin on the lattice $\vec{\theta}_i$ would be chosen, as well as a random two-dimensional unit vector $\vec{r}$. Then the clustering condition, rather than having the spins be the same, would be the condition $(\vec{\theta}_i \cdot \vec{r})(\vec{\theta}_i \cdot \vec{r}) > 0$, after which the spin would be accepted to cluster with some probability. The cluster would then be flipped by reflecting every spin in the cluster through the line perpendicular to $\vec{r}$. Such a clustering algorithm implementation would greatly speed up the 2D XY model simulation at low temperatures, enabling the simulation to reach an equilibrium configuration faster than the Metropolis algorithm used.

CONCLUSION

HOW TO IMPROVE CODE FOR FUTURE (efficiency, flexible equilibration times, correlation times etc.)

ACKNOWLEDGMENTS

We gratefully acknowledge the assistance of our teaching assistant Edwin Huang, who advised this project, answered questions, reviewed our code, and everything in between. We also thank Professor Blas Cabrera for his teaching and his work in putting together the new iteration of PHYSICS 113.

APPENDIX