1 Introduction

The Metropolis algorithm is a method which can be used for numerically simulating a 2D Ising model. The essential idea is that we begin with a random lattice of spins, either $+1$ or -1, and allow them to thermally equilibrate. We can simulate this by picking a random spin, and seeing if we should flip it or not. If the energy change associated with flipping the spin is negative, we accept the flip because we wish to move towards an energy minimum.

If the energy associated with the flip is positive, on the other hand, we need to be careful. According to statistical mechanics, that probability associated with this flip is the Boltzmann factor $P = e^{-\beta\Delta E}$.

So we randomly generate a number between zero and one, and check whether flipping the spin will change the energy by an amount higher or lower than the Boltzmann factor of that associated energy change. If the random number happens to be smaller than the Boltzmann factor, we accept the new configuration (this simulates the process in nature of a certain spin spontaneously changing with a certain probability). If not, we just keep the old configuration.

One iteration of this process is called a Monte Carlo sweep. Once several (often thousands) of sweeps have been performed, we can look for steady state behaviour of certain observables, like the energy

$$E = \frac{\partial \ln Z}{\partial \beta}$$

The magnetization,

$$M = \frac{1}{\beta}\frac{\partial \ln Z}{\partial h}$$

The heat capacity at constant volume,

$$C_V = \left(\frac{\partial E}{\partial T}\right)_V$$

And the magnetic susceptibility

$$\chi = -\frac{1}{N}\frac{\partial M}{\partial h}$$

We can get better results if we treat these quantities as statistical random variables, and perform several iterations of Monte Carlo simulations in order to determine the expectation values of these quantities, which are given by the usual formula

$$\langle \mathcal{O} \rangle := \sum_i \mathcal{O}_i P(\mathcal{O}_i)$$

2 Analytic Solutions for a 2×2 Lattice

We compute several quantities of interest: the energy E of the lattice, the magnetization M, the specific heat at constant volume C_V, and the magnetic susceptibility χ.

First, all possible configurations of spins and their associated energies are:

$$\begin{pmatrix} \uparrow & \uparrow \\ \uparrow & \uparrow \end{pmatrix} \qquad E = -8J - 4h$$

$$\begin{pmatrix} \downarrow & \uparrow \\ \uparrow & \uparrow \end{pmatrix} \begin{pmatrix} \uparrow & \downarrow \\ \uparrow & \uparrow \end{pmatrix} \begin{pmatrix} \uparrow & \uparrow \\ \downarrow & \uparrow \end{pmatrix} \begin{pmatrix} \uparrow & \uparrow \\ \uparrow & \downarrow \end{pmatrix} \qquad E = -2h$$

$$\begin{pmatrix} \downarrow & \downarrow \\ \uparrow & \uparrow \end{pmatrix} \begin{pmatrix} \uparrow & \downarrow \\ \uparrow & \downarrow \end{pmatrix} \begin{pmatrix} \uparrow & \uparrow \\ \downarrow & \downarrow \end{pmatrix} \begin{pmatrix} \downarrow & \uparrow \\ \downarrow & \uparrow \end{pmatrix} \qquad E = 0$$

$$\begin{pmatrix} \downarrow & \uparrow \\ \uparrow & \downarrow \end{pmatrix} \begin{pmatrix} \uparrow & \downarrow \\ \downarrow & \uparrow \end{pmatrix} \qquad E = 8J$$

$$\begin{pmatrix} \uparrow & \downarrow \\ \downarrow & \downarrow \end{pmatrix} \begin{pmatrix} \downarrow & \uparrow \\ \downarrow & \downarrow \end{pmatrix} \begin{pmatrix} \downarrow & \downarrow \\ \uparrow & \downarrow \end{pmatrix} \begin{pmatrix} \downarrow & \downarrow \\ \downarrow & \uparrow \end{pmatrix} \qquad E = 2h$$

$$\begin{pmatrix} \downarrow & \downarrow \\ \downarrow & \downarrow \end{pmatrix} \qquad E = -8J + 4h$$

Where the energies are computed via $\mathcal{H}(\{s_i\}) = -J \sum_{\{i,j\}} s_i s_j - h \sum_i s_i$, and with the convention that $\uparrow$ represents a spin $s = 1$, and $\downarrow$ represents a spin $s = -1$. Thus the partition function is

$$Z = e^{\beta(8J+4h)} + 4e^{2\beta h} + 4 + 2e^{-8\beta J} + 4e^{-2\beta h} + e^{\beta(8J-4h)}$$

Using this we can compute analytical expressions for E, M, C_V, and χ in the 2×2 case using the equations in section 1:

$$E = -\frac{1}{ZN} \left[(8J + 4h)e^{\beta(8J+4h)} + 8he^{2\beta h} - 16Je^{-8\beta J} - 8he^{-2\beta h} + (8J - 4h)e^{\beta(8J-4h)} \right]$$

$$M = \frac{1}{\beta Z}\left[4\beta e^{\beta(8J+4h)} + 8\beta e^{2\beta h} + 8\beta e^{-2\beta h} + 4\beta e^{\beta(8J-4h)}\right]$$

$$\begin{aligned} C_V = \frac{1}{Nk_BT^2Z^2}(((8J+4h)^2e^{\beta(8J+4h)} + 16h^2e^{2\beta h} \\ - 128J^2e^{-8\beta J} - 16h^2e^{-2\beta h} + (8J-4h)^2e^{\beta(8J-4h)})Z - ((8J+4h)e^{\beta(8J+4h)} \\ + 8he^{2\beta h} - 16Je^{-8\beta J} - 8he^{-2\beta h} + (8J-4h)e^{\beta(8J-4h)})^2) \end{aligned}$$

$$\begin{aligned} \chi = \frac{1}{\beta N^2Z^2}((16\beta^2e^{\beta(8J+4h)} + 16\beta^2e^{2\beta h} + 16\beta^2e^{-2\beta h} \\ + 16\beta^2e^{\beta(8J-4h)})Z - (4\beta e^{\beta(8J+4h)} + 8\beta e^{2\beta h} + 8\beta e^{-2\beta h} + 4\beta e^{\beta(8J-4h)})^2) \end{aligned}$$

3 Numerical Simulation Using the Metropolis Algorithm

First, a visual animation of the system is shown in figure 1 for a 100×100 lattice.

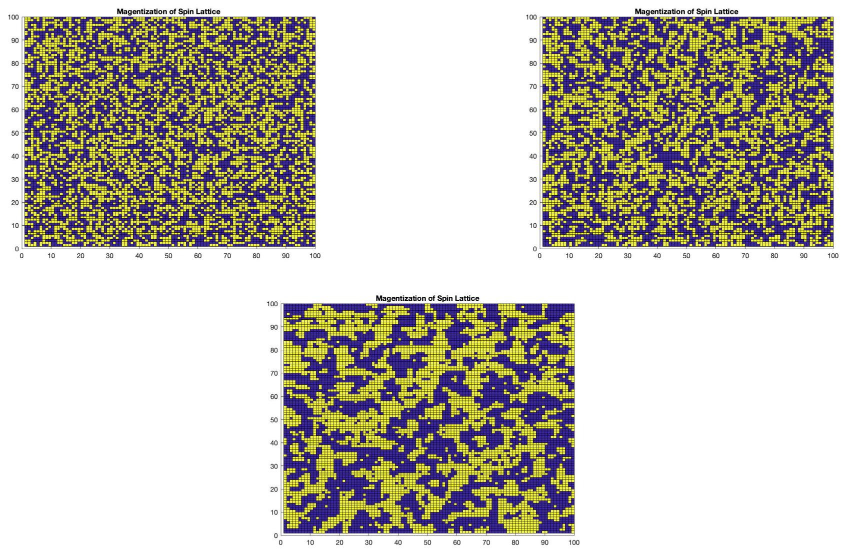

Figure 1: A 100 by 100 lattice of spins evolving through time with $J = 1$, $h = 0$, and $k_BT = 1$. The top left is at 1000 iterations, the top right is 5000, and the bottom is 20000. Small magnetic pockets are created.

For the remainder of the report we shall set $J = h = 1$ unless otherwise specified. We can check the validity of our scheme by comparing the analytic 2×2 solution with our numerical results. Setting the temperature, J, and h all to 1, we obtain the following after 50 Monte Carlo cycles:

Numerical Result after 50 Cycles			
E	M	C_V	χ
-2.999	0.999	0.003	7×10^{-5}

Analystical Result			
E	M	C_V	χ
-3	1	0	0

This is displayed in figure 2, where I have plotted the observables as a function of the number of Monte Carlo sweeps.

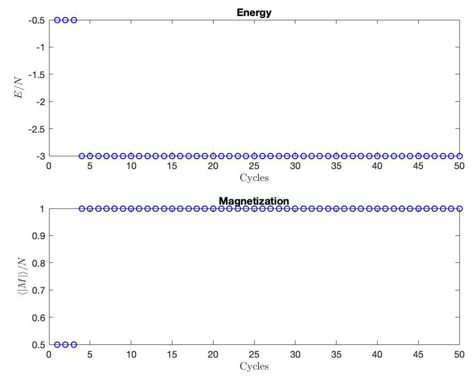

Figure 2: The observables with one Monte Carlo simulation (this is why C_V and χ are zero here; there is only one value of E and one value of M so there is no way to do statistics with them). E and M approach the desired values of -3 and 1, respectively.

However, we can do better than the plots shown in figure 2. In figure 3, we create the same plots, however this time we do 100 entire Monte Carlo simulations. For each simulation, we average the values of E and M at each number of cycles, creating a smoother plot. Furthermore, by doing this we have 100 values of E and M at each number of cycles, and hence we can calculate their variance in order to determine C_V and χ.

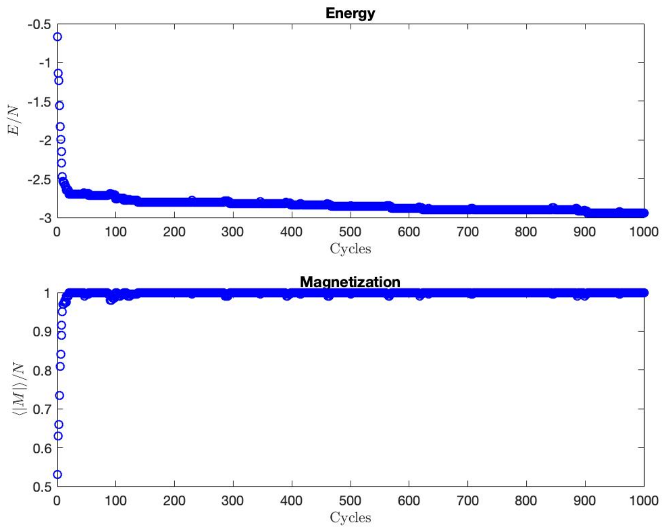

Figure 3: The observables, with each point on the graph averaged over 100 separate Monte Carlo simulations. Again we see the results going towards the expected analytical values.

Each quantity approaches a very close match with its numerical counterpart, indicating that the numerical results are trustworthy.

Finally, we investigate the behaviour of the observables as functions of temperature. In figure 4 we plot the four observables of interest. Unfortunately it was too computationally taxing to go above a 2×2 lattice without sacrificing grid spacing or the averaging process. However, the agreement with the analytical solutions from section 2 match very well with the simulation results.

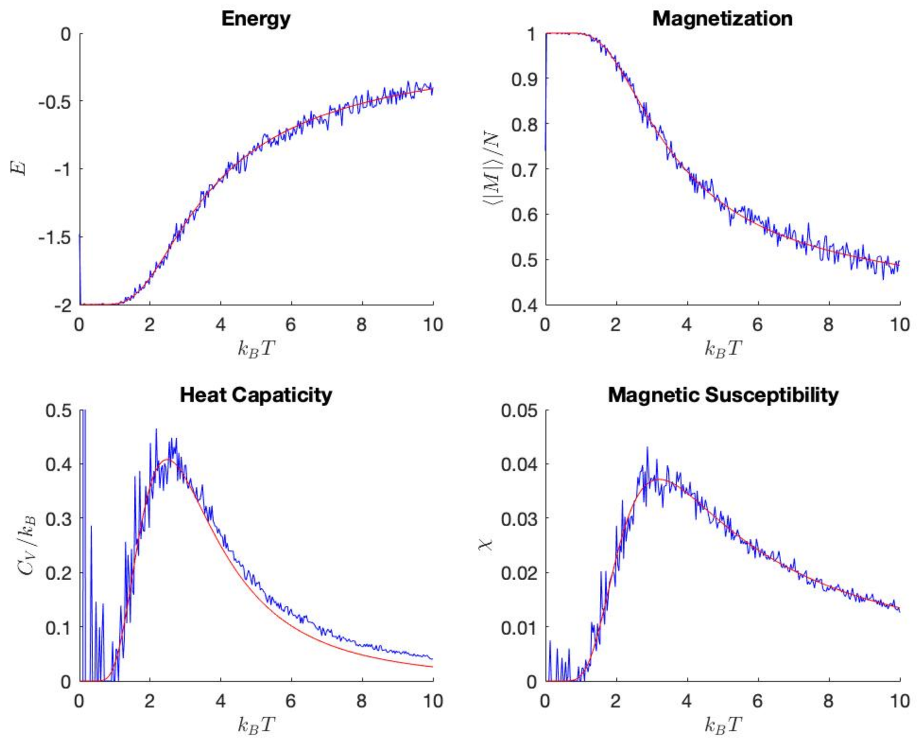

Figure 4: The observables versus Temperature in a 2×2 lattice. The red line is the analytic solution. The blue is the numerical result. 500 Monte Carlo simulations were performed, each with 40 iterations.

Now we turn our attention to a larger 20×20 system. In an attempt to determine the minimum number of Monte Carlo cycles after which the system is in equilibrium, we plot in figures 5 and 6 the same observables at temperatures $k_BT = 1$ and $k_BT = 2.4$, respectively. In both plots, we start with a random lattice of spins, and use $J = h = 1$.

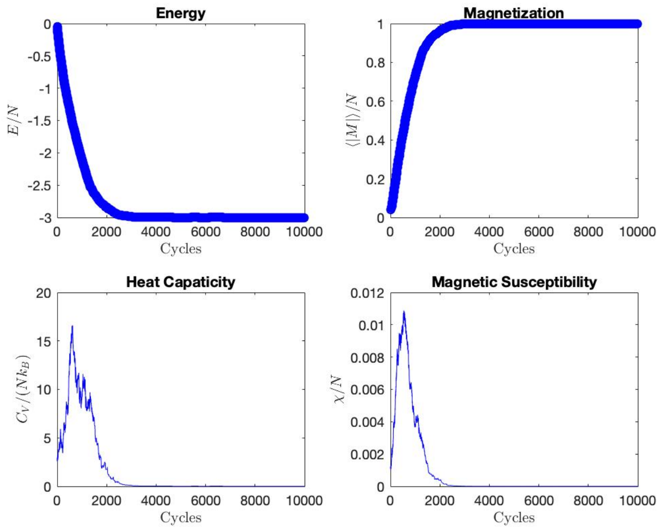

Figure 5: The number of Monte Carlo cycles it takes for the 20×20 lattice to reach a steady state at temperature $k_BT = 1$ is approximately 3000.

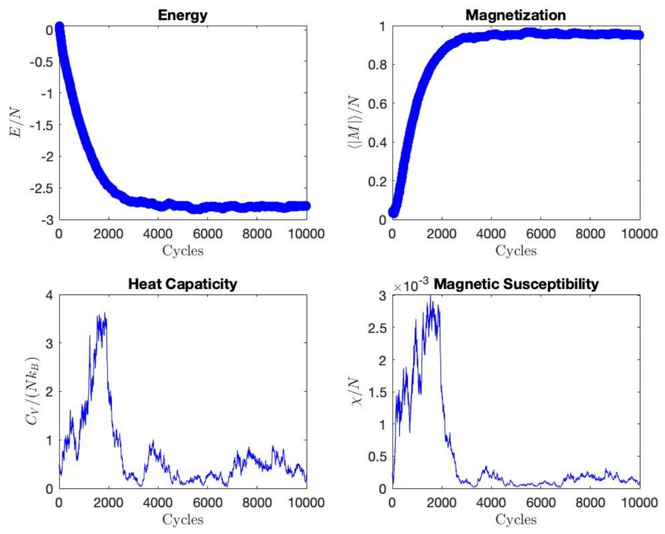

Figure 6: The number of Monte Carlo cycles it takes for the 20×20 lattice to reach a steady state at temperature $k_BT = 2.4$ is approximately 3000. But notice that this isn't a true steady state; there are still fluctuations in all of the observables. This is to be expected at higher temperatures.

We see that, as expected, the energy decays to a minimum and the magnetization goes to 1, since there is a magnetic field and the temperatures are sufficiently low (although in the $k_BT = 2.4$ plot we can already see some fluctuations). In figures 7 and 8, we construct the same plots, but instead of starting with a random lattice of spins, we start with all of the spins anti-aligned with the magnetic field. It takes significantly longer for the system to reach its steady state, as one would expect.

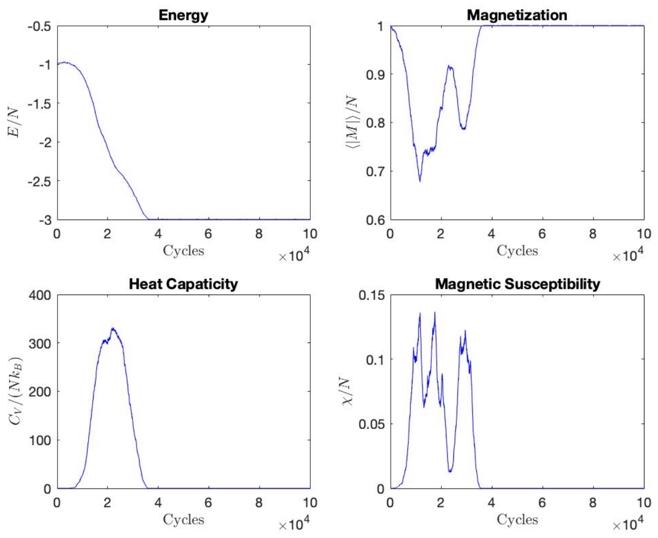

Figure 7: The system takes approximately 40000 Monte Carlo cycles to equilibrate at temperature $k_BT = 1$ when all the spins begin anti-aligned with the magnetic field. Notice that the spins all start anti-aligned so the absolute value of the magnetization must be 1. As they start flipping to align with the magnetic field, the absolute value of magnetization fluctuates, until eventually they all align, which is demonstrated by the curve returning to 1.

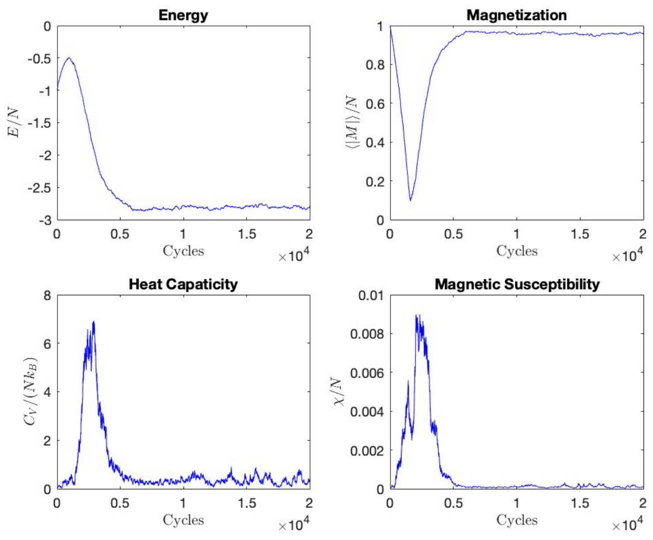

Figure 8: The system actually settles down faster at a higher temperature when the spins all begin anti-aligned to the magnetic field. It only takes about 5000 Monte Carlo cycles at $k_BT = 2.4$ for the system to settle (again, it does not reach a true equilibrium state as there are fluctuations in the observables). Again we see the magnetization start at 1, fluctuate while the spins start flipping, and then return to 1 as they start to align with the magnetic field.

Since starting with anti-aligned spins takes far more cycles, we will restrict ourselves to the case of randomly generated spins and conjecture that they reach equilibrium after 4000 cycles, to be on the safe side.

We can also consider the number of accepted configurations as a function of the number of cycles. The cases $T = 1$ and $T = 2.4$ are plotted in figure 9.

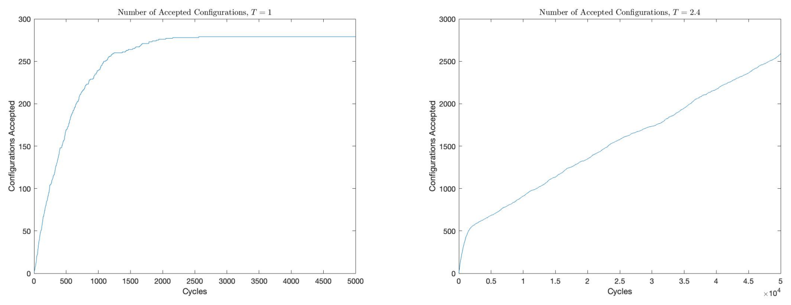

Figure 9: The number of accepted configurations by the k^{th} cycle.

The number of accepted configurations for low temperatures tends to level off to a constant value, while the accepted configurations for a higher temperature tends to increase linearly after a certain number of cycles. This makes sense; in the low temperature regime, the spins can align with the magnetic field $h = 1$, where it is unlikely that a spin will be flipped again because the system has reached equilibrium (see figure 5). On the other hand in the high temperature regime, the spins tend to align with the field but still fluctuate, allowing more configurations (see figure 6).

Now we can compute the probability $P(E)$ of the previous system with these temperatures. In figure 10, we see a histogram of the energies and their corresponding probabilities of occuring at $k_B T = 1$. A similar histogram is shown in figure 11, but with $k_B T = 2.4$. To create these histograms, we ran 100 Monte Carlo simulations, each performing 6000 sweeps. We only kept from 4000 sweeps onward to ensure that we had allowed the system to stabilize.

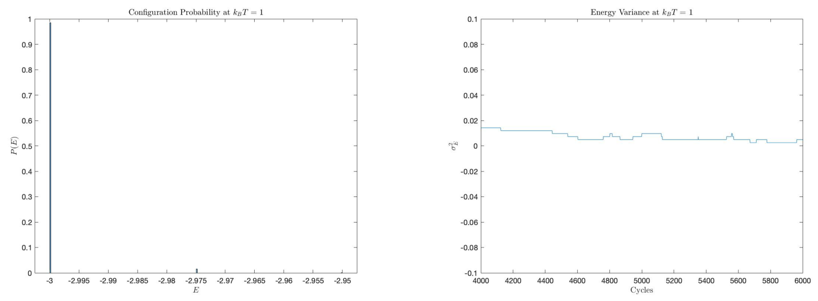

Figure 10: One the left is the histogram of energy probabilities at $k_BT = 1$. Almost all energies are precisely -3. On the right is the plot of the variance (which is essentially zero). In this regime of large numbers of cycles, the variance agrees with the histogram.

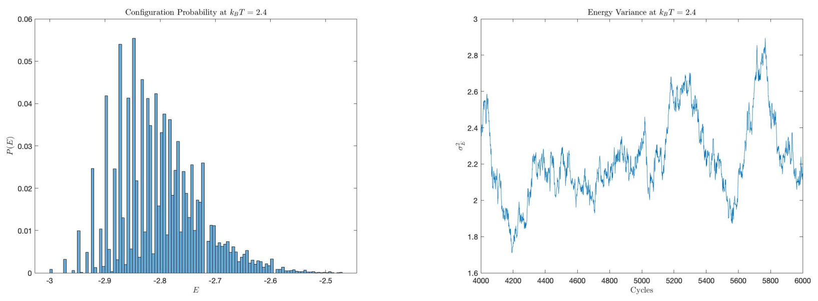

Figure 11: The histogram of energy probabilities at $k_BT = 2.4$ is on the left. The energies are far more spread out than they were at $k_BT = 1$; in fact, they look to be approaching a normal distribution. Notice that on the right, the variance of the energy in this cycle regime corresponds nicely to the histogram (the energy fluctuates according to both the histogram and σ_E^2.

4 MATLAB Code

```
clear;clc

%Simulation parameters
L = 100; M = 100; J = 1; h = 0; kT = [1]; %linspace(0,10,100)
    ;
```

```
N = L*M; %Number of spins
sweepsPerSimulation = 50000; %Iterate a large amount more
    than the number of spins
numberOfSimulations = 1;

%Define lots of empty arrays/matrices
E_static = zeros(numberOfSimulations,length(kT));
M_static = zeros(numberOfSimulations,length(kT));
Chi = zeros(1,length(kT));
C = zeros(1,length(kT));

E_transient = zeros(numberOfSimulations,sweepsPerSimulation);
M_transient = zeros(numberOfSimulations,sweepsPerSimulation);
C_transient = zeros(1,sweepsPerSimulation);
Chi_transient = zeros(1,sweepsPerSimulation);

acceptedConfigs = zeros(1,sweepsPerSimulation); counter = 0;


for l = 1:length(kT) %Perform Monte Carlo at each temperature

    beta = 1/kT(l);

    for p = 1:numberOfSimulations %Perform certain amount of
    simulations

        %Begin Monte Carlo
        x = 1:L; y = 1:M; [X,Y] = meshgrid(x,y); %Mesh grid
    for plotting later

        %Startw with empty lattice
        Snew = zeros(L,M);

        %Fill the lattice with random spins
        for i = 1:L
            for j = 1:M
                Snew(i,j) = RS;
            end
        end

        for k = 1:sweepsPerSimulation
            %Begin with an empty L by M lattice
            S = Snew;

            %Select a random spin from the lattice
            n = randi(L,1); m = randi(M,1);
            s = S(n,m);

            Eb = Espin(S,n,m,L,M,J,h); %Energy in this
   configuration of spins

            %Change the spin s_i |--> -s_i
            Strial = S;
            Strial(n,m) = -s;

            %Calculate the energy from the trial
   configuration and
            %the change in energy from the old configuration
            Et = Espin(Strial,n,m,L,M,J,h); %trial energy
            dE = Et - Eb; %Change in energy due to this flip

            %See if the new lattice is acceptable of not
            Pacc = exp(-beta * dE); r = rand;
            if (dE < 0) || (r < Pacc)
                Snew = Strial; %Keep the new lattice, keep dE
                counter = counter+1;
                acceptedConfigs(k) = counter;
            else
                Snew = S;
                dE = 0; %Keep the old lattice, dE must be
   zero
                acceptedConfigs(k) = counter;
            end

            %Animation

            if rem(k, 1000)==0
                figure(1)
                hold on
                pcolor(X,Y,Snew);
                drawnow;
                title('Magentization of Spin Lattice')
            end


            %Only measure the observables after they have had
    a chance
            %to equilibrate (i.e. after the desired number of
    cycles)
            if k == sweepsPerSimulation
```

```
                E_static(p,l) = H(Snew,L,M,J,h); %Per spin
                M_static(p,l) = Mmean(Snew); %Per spin
            end

        %Transient part (Plot this vs. number of cycles)
        E_transient(p,k) = H(Snew,L,M,J,h);
        M_transient(p,k) = Mmean(Snew); %Mmean(Snew);

        end
    end
    %Calculate observables versus number of cycles
    Energy = mean(E_static);
    Magnetization = mean(M_static);
    C(l) = var(E_static(:,l)) .* beta^2; %Really C/k
    Chi(l) = var(M_static(:,l)) .* beta;
end

%For plotting C and Chi versus cycles
for k = 1:sweepsPerSimulation
    C_transient(k) = (mean(E_transient(:,k).^2) - mean(
   E_transient(:,k)).^2) * beta^2;
    Chi_transient(k) = (mean(M_transient(:,k).^2) - mean(
   M_transient(:,k)).^2) * beta;
end




%Plot number of accepted configurations vs cycles
%{
figure(1)
plot(1:sweepsPerSimulation, acceptedConfigs)
title('Number of Accepted Configurations, $T=1$', '
   Interpreter', 'LaTeX')
xlabel('Cycles')
ylabel('Configurations Accepted')
%}




%Plot probability of configuration
%{
E_stable = E_transient(:,4000:end);
figure(2)
histogram(E_stable./N, 'Normalization', 'probability')
xlabel('$E$', 'Interpreter', 'LaTeX')
ylabel('$P(E)$', 'Interpreter', 'LaTeX')
title(['Configuration Probability at $k_BT =$ ', num2str(kT)
    ], ...
    'Interpreter', 'LaTeX')

figure(3)
plot(4000:6000, var(E_stable)/N)
xlabel('Cycles', 'Interpreter', 'LaTeX')
ylabel('$\sigma_E^2$', 'Interpreter', 'LaTeX')
title(['Energy Variance at $k_BT =$ ', num2str(kT)], ...
    'Interpreter', 'LaTeX')
%}




%Plot observables vs cycles
%{
figure(4)
subplot(2,2,1)
plot(1:sweepsPerSimulation, mean(E_transient)./N, 'b')
title('Energy')
xlabel('Cycles', 'Interpreter', 'LaTeX')
ylabel('$E/N$', 'Interpreter', 'LaTeX')

subplot(2,2,2)
plot(1:sweepsPerSimulation, mean(M_transient), 'b')
title('Magnetization')
xlabel('Cycles', 'Interpreter', 'LaTeX')
ylabel('$\langle |M| \rangle / N$', 'Interpreter', 'LaTeX')

subplot(2,2,3)
plot(1:sweepsPerSimulation, C_transient./N, 'b')
title('Heat Capacity')
xlabel('Cycles', 'Interpreter', 'LaTeX')
ylabel('$C_V/(Nk_B)$', 'Interpreter', 'LaTeX')

subplot(2,2,4)
plot(1:sweepsPerSimulation, Chi_transient, 'b')
title('Magnetic Susceptibility')
xlabel('Cycles', 'Interpreter', 'LaTeX')
ylabel('$\chi/N$', 'Interpreter', 'LaTeX')
%}
```

```matlab





%Plot observables vs temperature
figure(5)
subplot(2,2,1)
plot(kT, Energy./N, 'b')
title('Energy')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$E$', 'Interpreter', 'LaTeX')

subplot(2,2,2)
plot(kT, Magnetization, 'b')
title('Magnetization')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$\langle |M| \rangle / N$', 'Interpreter', 'LaTeX')

subplot(2,2,3)
plot(kT, C./N, 'b')
title('Heat Capacity')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$C_V/k_B$', 'Interpreter', 'LaTeX')
ylim([0,2])

subplot(2,2,4)
plot(kT, Chi, 'b')
title('Magnetic Susceptibility')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$\chi$', 'Interpreter', 'LaTeX')






%Plot Numerical vs Analytical
%{
E_true = zeros(1,length(kT));
M_true = zeros(1,length(kT));
Cv_true = zeros(1,length(kT));
Chi_true = zeros(1,length(kT));
for i = 1:length(kT)
    beta = 1/kT(i);
    E_true(i) = E_analytic(beta,J,h,N);
    M_true(i) = M_analytic(beta,J,h,N);
    Cv_true(i) = Cv_analytic(beta,J,h,N);
    Chi_true(i) = Chi_analytic(beta,J,h,N);
end

figure(4)
subplot(2,2,1)
hold on
plot(kT, Energy./N, 'b')
plot(kT, E_true, 'r')
hold off
title('Energy')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$E$', 'Interpreter', 'LaTeX')

subplot(2,2,2)
hold on
plot(kT, Magnetization, 'b')
plot(kT, M_true, 'r')
hold off
title('Magnetization')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$\langle |M| \rangle / N$', 'Interpreter', 'LaTeX')

subplot(2,2,3)
hold on
plot(kT, C./N, 'b')
plot(kT, Cv_true, 'r')
hold off
title('Heat Capacity')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$C_V/k_B$', 'Interpreter', 'LaTeX')
ylim([0,0.5])

subplot(2,2,4)
hold on
plot(kT, Chi, 'b')
plot(kT, Chi_true, 'r')
hold off
title('Magnetic Susceptibility')
xlabel('$k_BT$', 'Interpreter', 'LaTeX')
ylabel('$\chi$', 'Interpreter', 'LaTeX')
ylim([0,0.05])
%}
```

```
%Analytic expression for the energy of the lattice (2x2 case)
function [result] = E_analytic(beta,J,h,N)

result = -(1/Z(beta,J,h)) * ((8*J+4*h)*exp(beta*(8*J+4*h)) + ...
     8*h*exp(2*beta*h) - 16*J*exp(-8*beta*J) - 8*h*exp(-2*beta*h) ...
     + (8*J - 4*h)*exp(beta*(8*J-4*h))) / N;

end

%Analytic expression for the mean magnetization of the lattice (2x2 case)
function [result] = M_analytic(beta,J,h,N)

result = 1/(beta * Z(beta,J,h)) * (4*beta*exp(beta*(8*J+4*h)) + ...
     8*beta*exp(2*beta*h) + 8*beta*exp(-2*beta*h) + ...
     4*beta*exp(beta*(8*J-4*h))) / N;

end

function [result] = Cv_analytic(beta,J,h,N)

result = (((8*J+4*h)^2*exp(beta*(8*J+4*h)) + ...
     16*h^2*exp(2*beta*h) - 128*J^2*exp(-8*beta*J) - 16*h^2*exp(-2*beta*h) ...
     + (8*J - 4*h)^2*exp(beta*(8*J-4*h))) * Z(beta,J,h) - ...
     ((8*J+4*h)*exp(beta*(8*J+4*h)) + ...
     8*h*exp(2*beta*h) - 16*J*exp(-8*beta*J) - 8*h*exp(-2*beta*h) ...
     + (8*J - 4*h)*exp(beta*(8*J-4*h)))^2) / Z(beta,J,h)^2 * beta^2/N;

end

function [result] = Chi_analytic(beta,J,h,N)

result = ((16*beta^2*exp(beta*(8*J+4*h)) + ...
     16*beta^2*exp(2*beta*h) + 16*beta^2*exp(-2*beta*h) + ...
     16*beta^2*exp(beta*(8*J-4*h)))*Z(beta,J,h) - ...
```

```
    ((4*beta*exp(beta*(8*J+4*h)) + ...
    8*beta*exp(2*beta*h) + 8*beta*exp(-2*beta*h) + ...
    4*beta*exp(beta*(8*J-4*h))))^2 ) / (beta * N^2 * Z(beta,J
   ,h)^2);

end

%Analytic partition function (2x2 case)
function [result] = Z(beta,J,h)

result = exp(beta*(8*J+4*h)) + 4*exp(2*beta*h) + 4 + 2*exp
   (-8*beta*J) ...
    + 4*exp(-2*beta*h) + exp(beta*(8*J-4*h));

end

%Hamiltonian, J * sum_{i,j}(s_i*s_j) - h * sum_i(s_i)
function [result] = H(S,L,M,J,h)

%Sum over "nearest neighbours" at the spin S(n,m). Recall
   that
%   a = n-1, b = n+1, c = m-1, d = m+1.
neighbours = @(n,m) S(n,m)*S(a(n,L),m) + S(n,m)*S(b(n,L),m) +
    ...
    S(n,m)*S(n,c(m,M)) + S(n,m)*S(n,d(m,M));

%Sum over all nearest neighbors for each spin in the lattice
%and sum over all spins. The factor of 1/2 accounts for the
   double counting.

P = 0; G = 0;
for i = 1:L
    for j = 1:M
        P = P + neighbours(i,j);
        G = G + S(i,j);
    end
end

result = -1/2 * J*P - h*G;

end

%Energy of a single spin S(n,m)
function [result] = Espin(S,n,m,L,M,J,h)

```

```matlab
%Sum over "nearest neighbours" at the spin S(n,m). Recall that
%   a = n-1, b = n+1, c = m-1, d = m+1.
%The factor of 1/2 accounts for the double counting.
neighbours = (S(n,m)*S(a(n,L),m) + S(n,m)*S(b(n,L),m) + ...
    S(n,m)*S(n,c(m,M)) + S(n,m)*S(n,d(m,M)));

G = S(n,m);

result = -J*neighbours - h*G;

end

%Mean magnetization
function [result] = Mmean(S)

result = abs(mean(mean(S)));

end

%The following four functions allow me to implement the periodic boundary
%conditions
function [result] = a(n,L)
if n == 1
    result = L;
else
    result = n-1;
end
end
function [result] = b(n,L)
if n == L
    result = 1;
else
    result = n+1;
end
end
function [result] = c(m,M)
if m == 1
    result = M;
else
    result = m-1;
end
end
function [result] = d(m,M)
```

```
if m == M
    result = 1;
else
    result = m+1;
end
end

%Random spins, either -1 or 1
function [result] = RS
x = rand;
if x<0.5
    result = -1;
else
    result = 1;
end
end
```