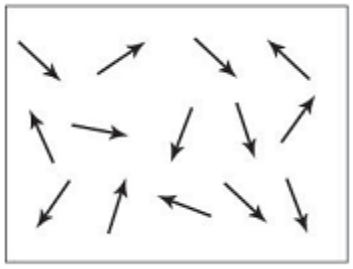
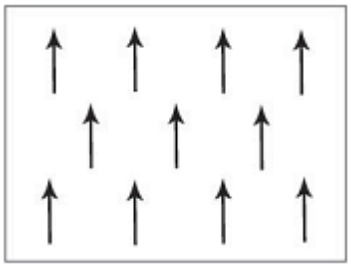
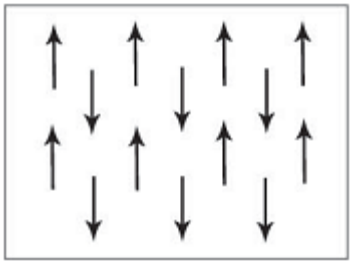
EEE 424: Numerical Methods Laboratory

Lab Group: A1

"Monte-Carlo" Simulation of Magnetization in Ferromagnetic Material with Temperature Variation







Submitted to-

Md. Imran Momtaz
Assistant Professor
Dept. of EEE
BUET

Soumitra Roy Joy
Lecturer
Dept. of EEE
BUET

Submitted by-

Abul Hasan Fahad
Std. No.: 0806021

Monjurul Feeroz Meem
Std. No.: 0806028

Project Proposal:

New Project Proposal (Lab Group 1)

Course No.: EEE 424

Course title: Numerical Methods Laboratory

Name of the project:

Simulation of Magnetism of Ferromagnetic Material with variation of temperature using Monte Carlo Simulation Method

Description:

In this project we want to simulate the magnetic behavior of ferromagnetic materials. Ferromagnetic material is composed of tiny magnetic domains, each having a magnetic dipole moment. Each magnetic dipole is under influence of adjacent dipoles. Thermal energy from environment causes the changes in directional dipole moment (spin) and as a result magnetization of the material changes. So, we wish to vary the temperature of the system and want to see how these dipoles behave collectively in our computer program. The mathematical model which is required for our purpose is known as "Ising Model". To implement the Ising Model- we resort to Monte Carlo Simulation method. This technique will be employed in such a way so that the simulation resembles the randomness associated with actual materials upto a satisfactory level.

Submitted by-

Abul Hasan Fahad (0806021)

Monjurul Feroz Meem (0806028)

“Monte-Carlo” Simulation of Magnetization in Ferromagnetic Material with Temperature Variation:

- **Ising Model:** To simulate magnetic behavior of a Ferromagnetic Material subject to temperature variation, we resort to the “Ising” model, proposed by physicist Ernst Ising who did it in 1920s. It is a mathematical model in statistical mechanics. It has since been used to model diverse phenomena in which bits of information, interacting in pairs, produce collective effects. Serious applications include complicated models of ferromagnets, fluids, alloyds, interfaces, nuclei, subnuclear particles. Now-a-days, it is one of the pillars of Computational physics.

- **Test System:** In this project, we have used it to model the behavior of a 2D ferromagnetic domain. The domain can be conceived as a “Grid of Magnetic Dipoles”. This grid is of size 10-by-10, having 100 dipoles. We assume the dipoles either have “spin up” or “spin down”. For spin-up (↑) we have $s_i = 1$ and spin-down (↓) we have $s_i = -1$.

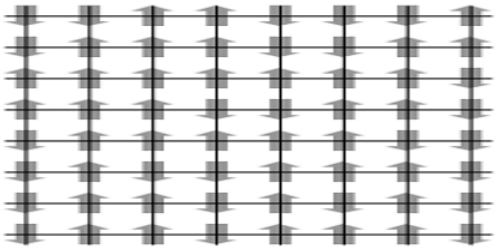


Figure: 2D Ising Grid

- **Interaction Energy:** Magnetism in the material depends on the interaction of adjacent dipoles. The energy due to interaction of adjacent dipoles: −J if parallel and +J if anti-parallel.

  J is assigned a value "> 0" (greater than zero) for ferromagnets.

- **Energy of a configuration:** Energy of the dipole-arrangement is calculated by the following expression:

$$E = -J \sum_{<kl>}^{N} s_k s_l - B \sum_{k}^{N} s_k$$

Where, N= Total no. of spins

<kl>= sum of nearest members (periodic boundary condition)

B= external Magnetic field

In our case, we have excluded external magnetic field. So, we employed the expression-

$$E = -J \sum_{<kl>}^{N} s_k s_l$$

- **Magnetization of a configuration:** Magnetization is calculated as-

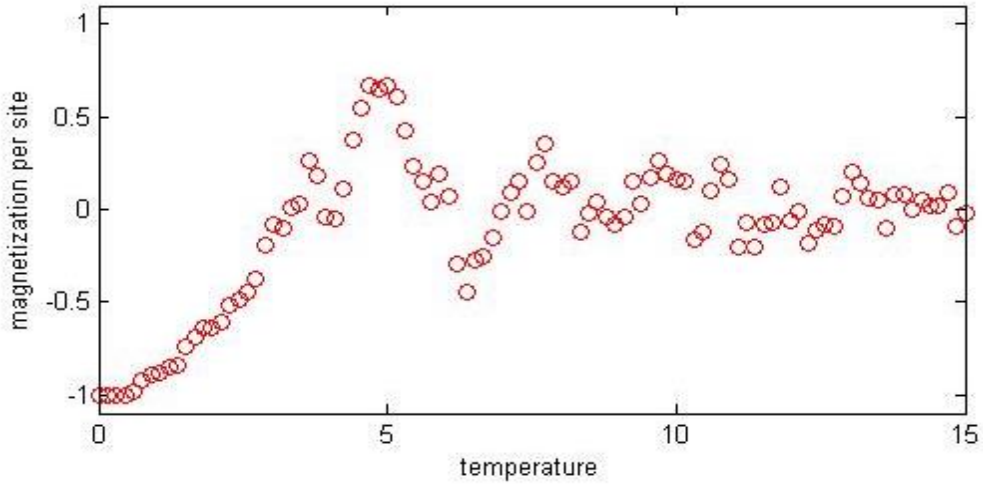
$$\mathcal{M}_i = \sum_{j=1}^{N} s_j$$

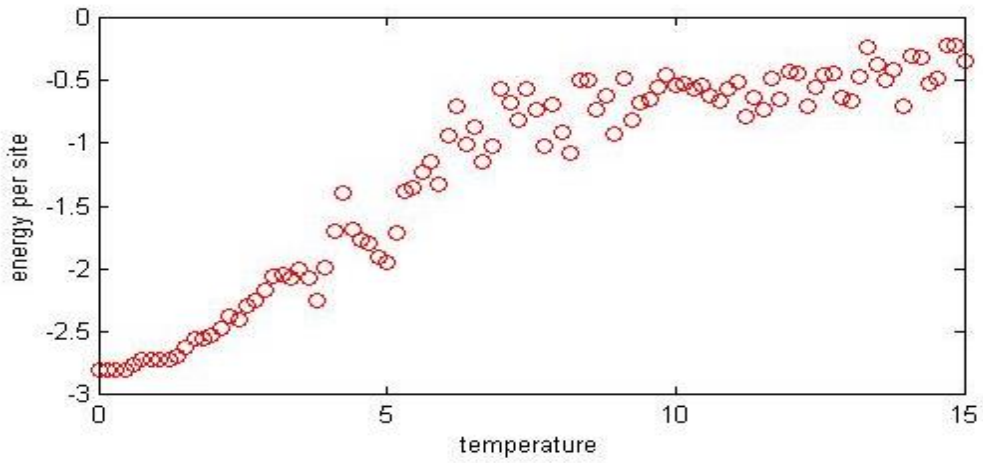
Where M_i= sum of all the spins in "i-th" configuration

"Metropolis Monte-Carlo Algorithm" to simulate Ising Model:

1. Calculate energy of the lattice at a given temperature.
2. Choose a random position in the lattice. Flip its spin and calculate the present energy of the system.
3. Take the difference between present and previous energy.
4. If the energy is positive then keep the spin flipped.
5. Otherwise, calculate- $w = e^{-(\beta \Delta E)}$, where β = 1/KT, ΔE= energy difference. Assume K=1, so that T becomes dimensionless.
6. Compare w with a random number r. If r<=w, accept the spin-flip, otherwise keep the initial system.
7. Perform this Monte-Carlo Loop for, at least 100 times. Calculate energy and magnetization at each step.
8. After the looping is done, take the average of energy and magnetization. This is our result for a specific temperature.
9. Vary the temperature and find the value of energy and magnetization at each temperature.
10. Also vary interaction energy and plot the energy and magnetization

Results:





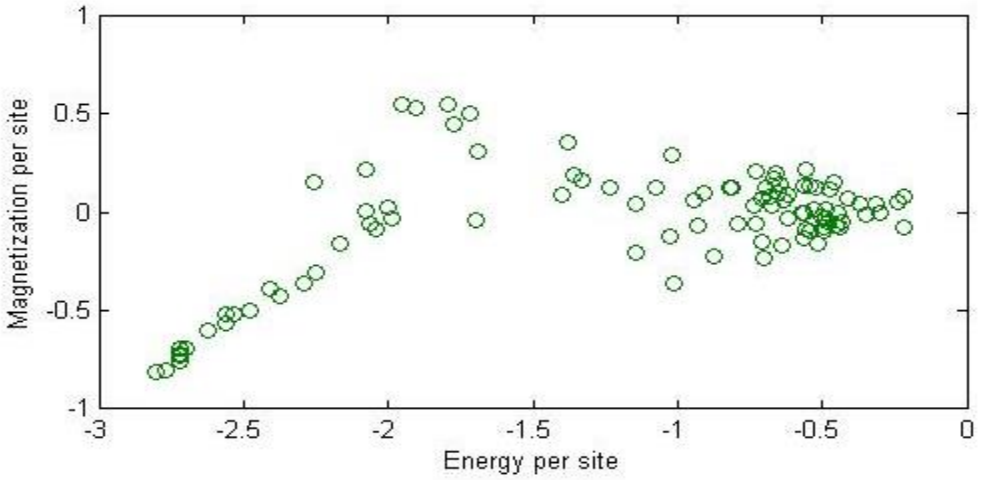
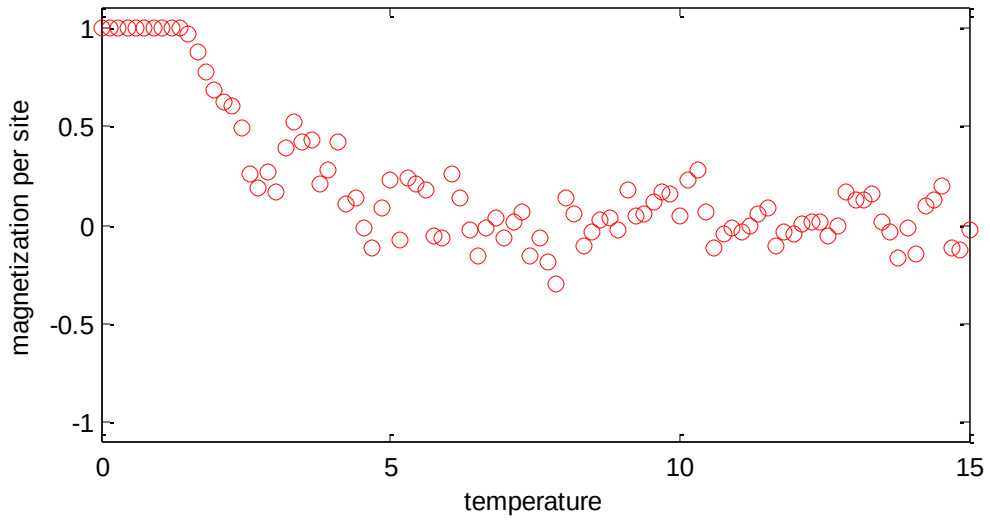
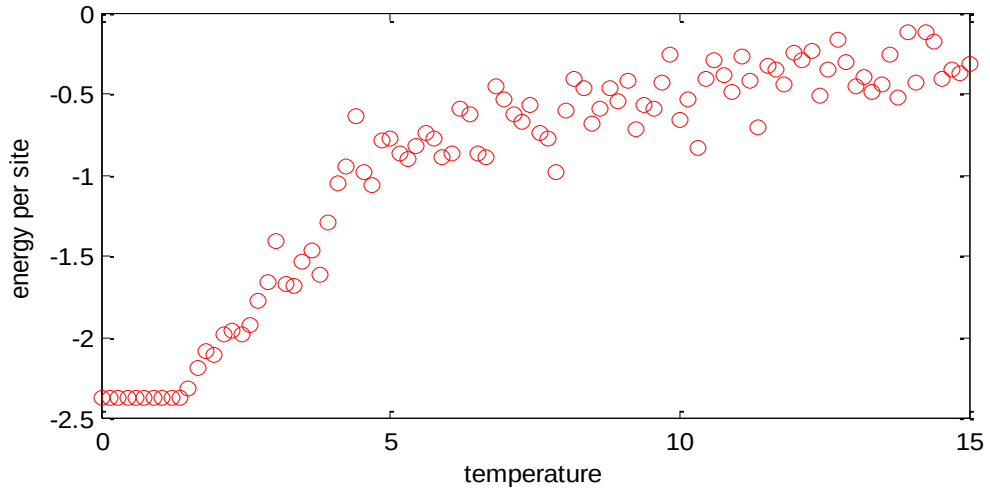


Figure: Results for Interaction Energy J=1





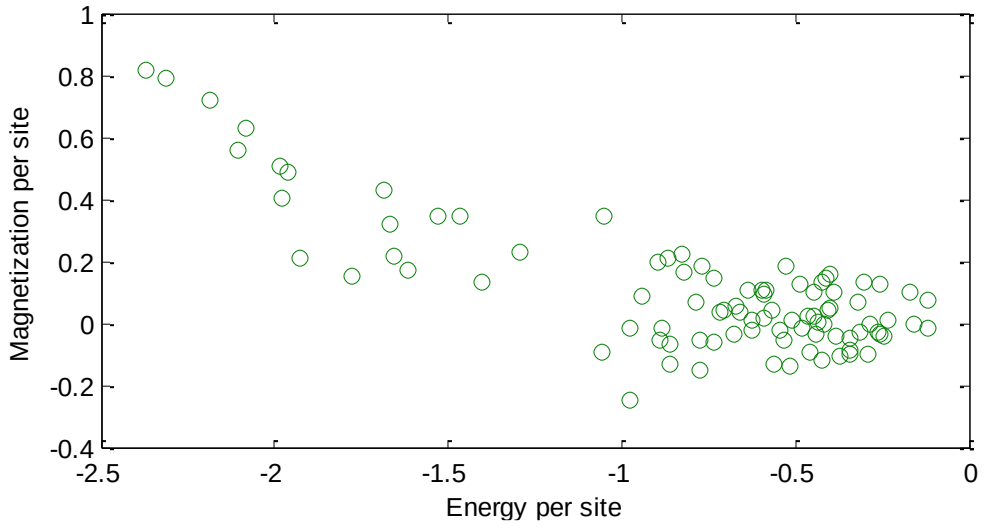
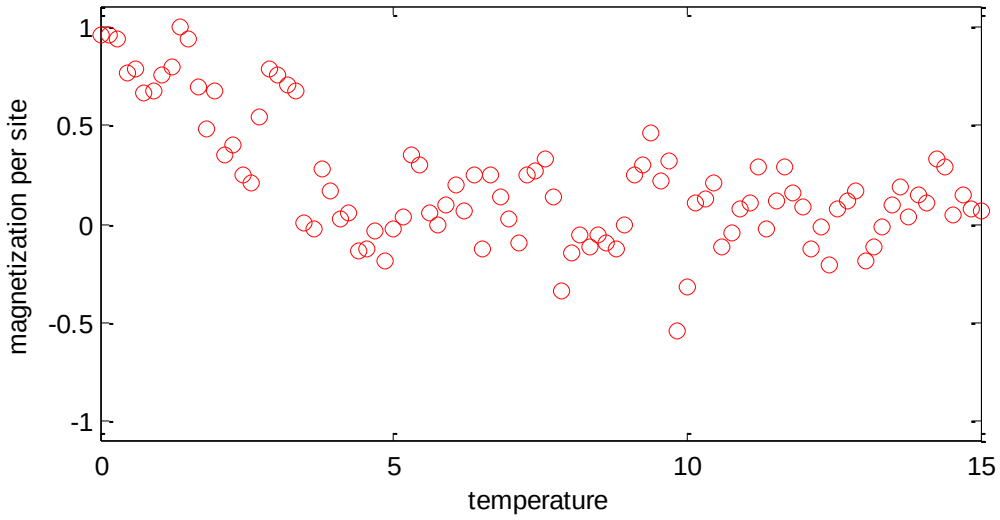
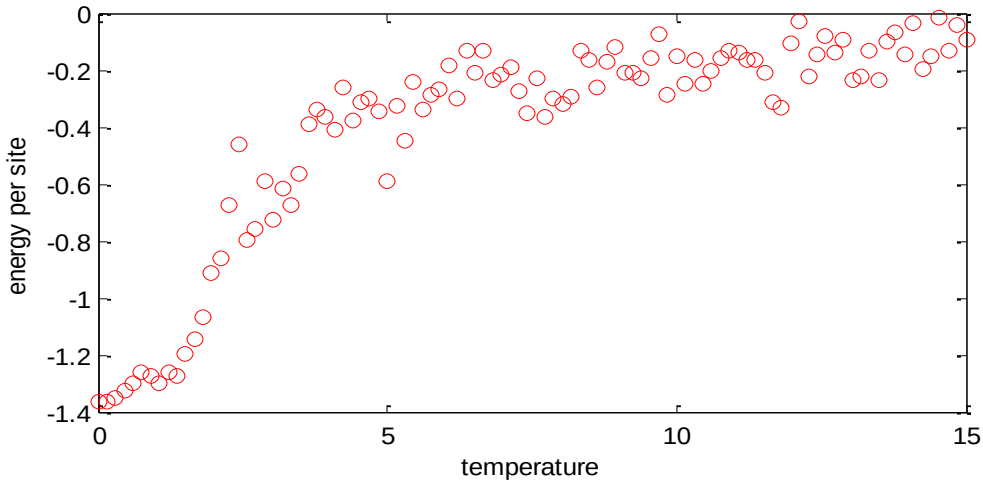


Figure: Results for Interaction Energy J=0.8





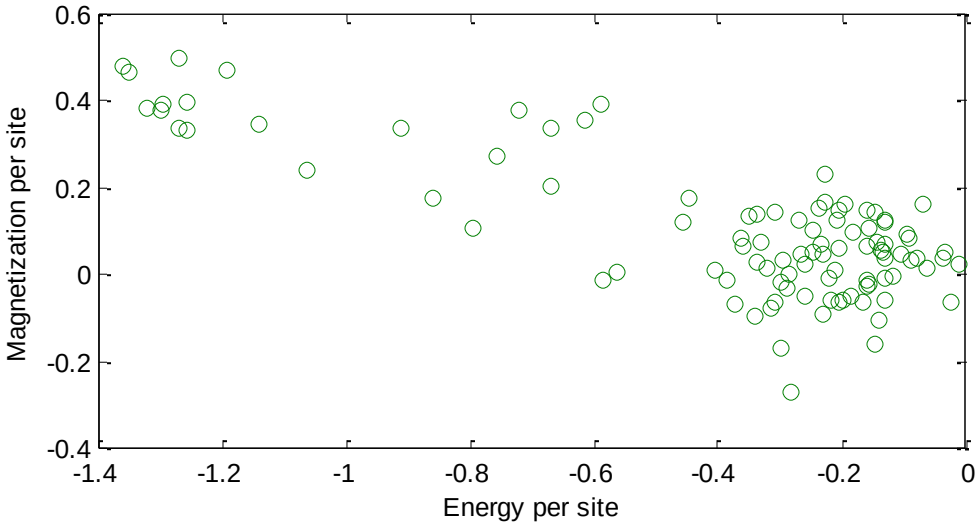


Figure: Results for interaction energy J=0.5

Magnetization

(Y axis)

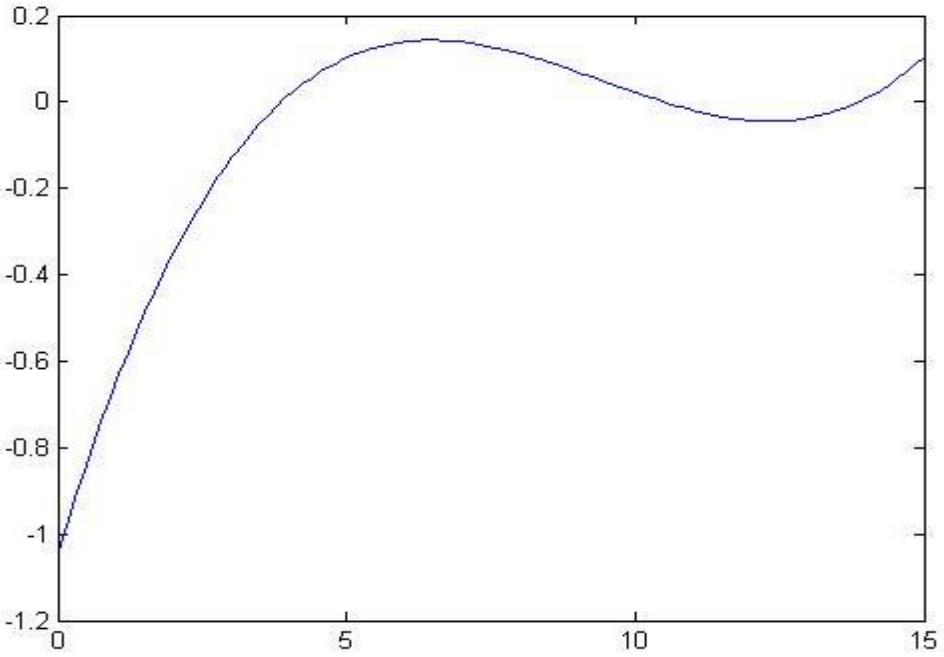


Figure: Magnetization vs. Temperature fit for J=1.0

Temperature (X axis)

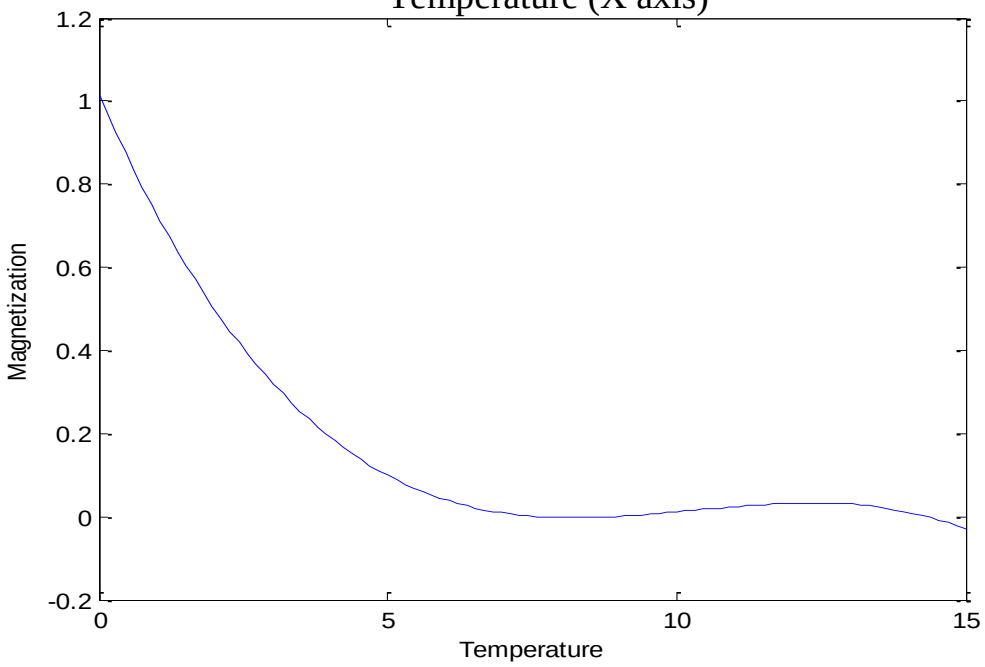


Figure: Magnetization vs. Temperature fit for J=0.8

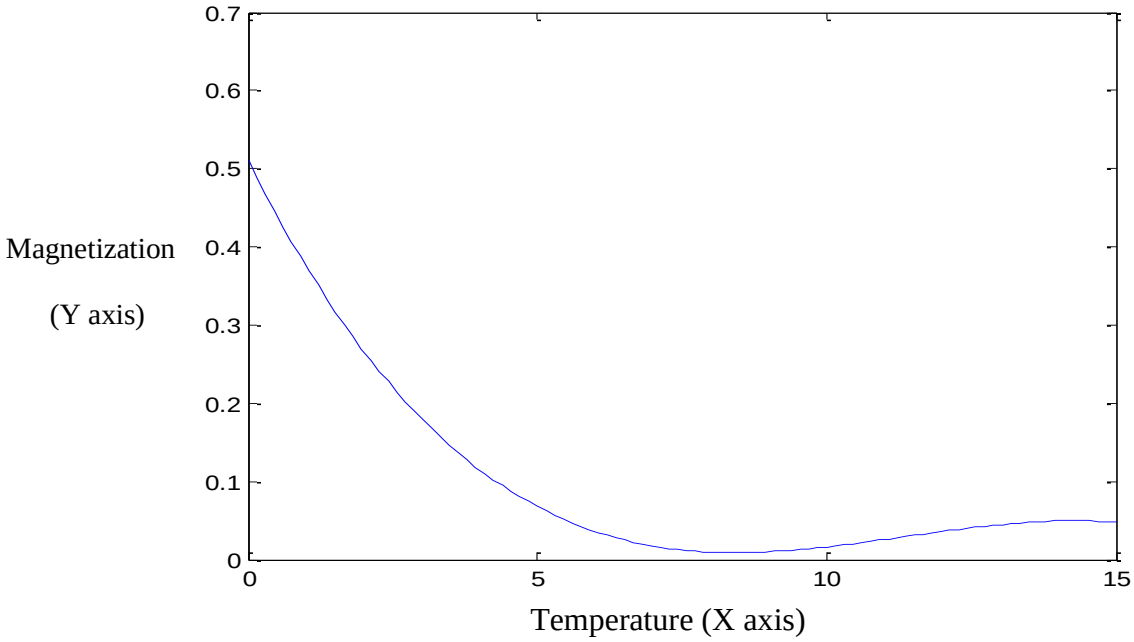


Figure: Magnetization vs. temperature 3rd order fit for J=0.5

Discussion:

1. Ising Model is a mathematical model, which employs statistical techniques to model the behavior of ferromagnetism. It is not a physical model. So it can't predict real values of transition temperatures for different materials.
2. From the obtained Magnetization vs. Temperature curves, it is clear that there is a ferromagnetic to paramagnetic transition between T=0 to T=5.
3. System Energy increases with temperature, as expected.
4. As evident from our results, Magnetization increases as energy decreases.
5. As interaction energy decreases, the phase transition occurs earlier. It is expected as dipoles are weakly interacting.
6. If sufficient time is provided, we could have extended our work to 3D system.

Code:

```
clear all;
close all;
clc;
n = 10;
b = -ones(n,n);
b = b .^ round(rand(n,n));
save('A_load','b')
```

```
clc; close all;
n = 10;
K = 1;
% K=1.38e-23;
J = 1;
B = 0;
temp_point = 50;
monte_loop = 200;
% a = -ones(n,n);
% a = a .^ round(rand(n,n));
load('A_load','b');
a=b;
figure(1)
image((a+1)*128)
axis square; colormap bone;drawnow
T = linspace(15,0,temp_point);

for t = 1:length(T)
    beta = 1/(K*T(t));
   for g = 1:monte_loop
        i = floor((n-1).*rand(1) + 1);
        j = floor((n-1).*rand(1) + 1);
        eb = -J*energy(a)-B*sum(sum(a));%(energy before change)
        a(i,j) = -1*a(i,j);
        et = -J*energy(a)-B*sum(sum(a));%(energy after change)
        dele = (et-eb);
        if dele > 0
            w = exp(-beta*dele);
            r = rand();
            if r > w
                a(i,j) = -1*a(i,j);
            end
        end
        Mag(g) = sum(sum(a));
        En(g) = -J*energy(a)-B*sum(sum(a));
        figure(2)
        image((a+1)*128)
        xlabel(sprintf('T = %0.2f, M = %0.2f, E = %0.2f', T(t), Mag(g)/n^2,
En(g)/n^2));
        set(gca,'YTickLabel',[],'XTickLabel',[]);
        axis square; colormap bone;drawnow
    end
```

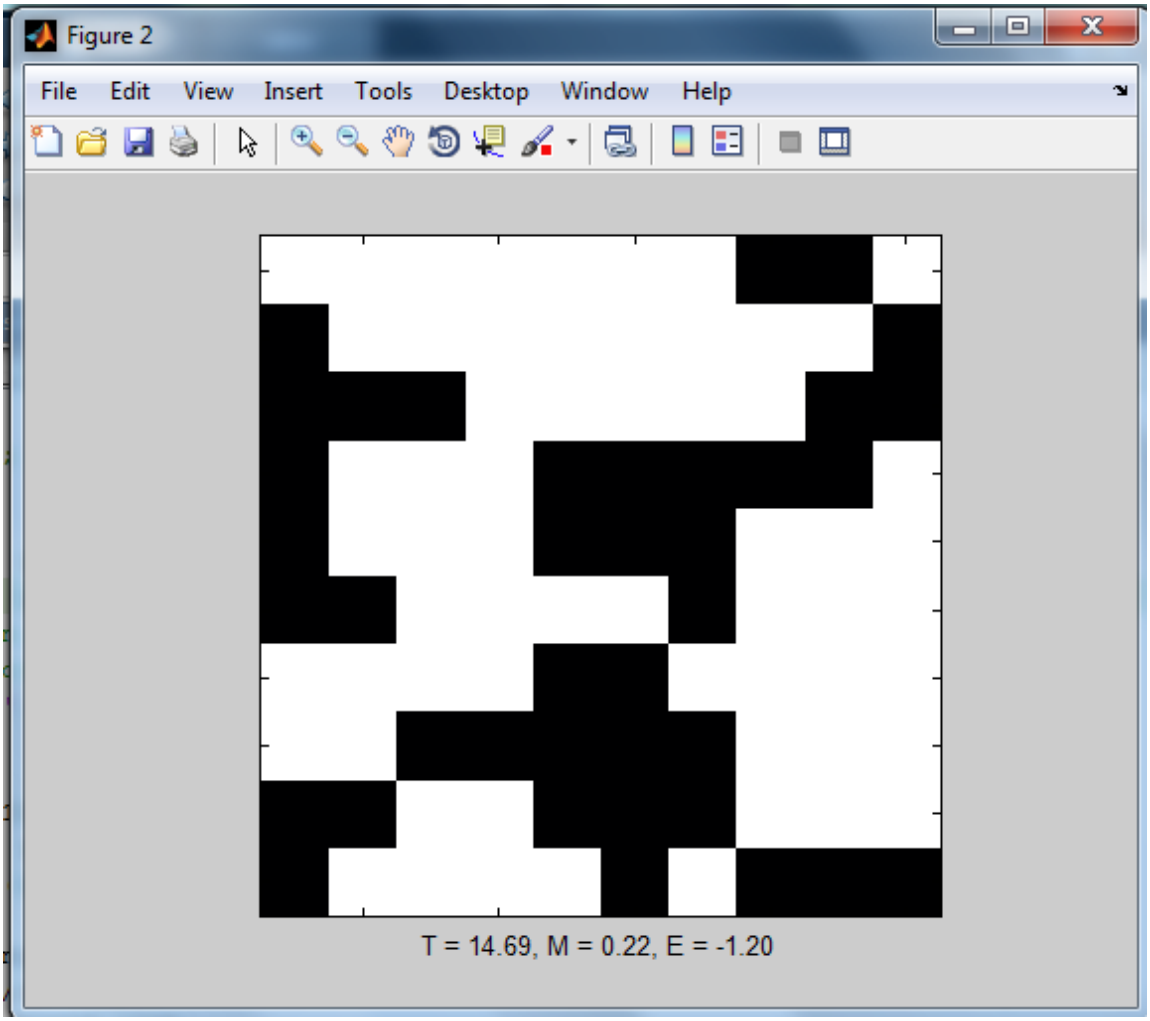
```
    M(t) = mean(Mag)/(n*n);
    E(t) = mean(En)/(n*n);

end
%magnetization
figure()
plot(T,M/max(abs(M)),'ro')
ylabel('magnetization per site');
xlabel('temperature');
ylim([-1.1 1.1]);
pbaspect([2 1 1]);
print(gcf, '-depsc2', 'ising-magnetization');
%energy
figure()
plot(T, E, 'ro');
ylabel('energy per site');
xlabel('temperature');
pbaspect([2 1 1]);
print(gcf, '-depsc2', 'ising-energy');
% Magnetization per site, versus Energy per site
figure()
plot(E, M, 'o', 'Color', [0 0.5 0]);
xlabel('Energy per site');
ylabel('Magnetization per site');
pbaspect([2 1 1]);
print(gcf, '-depsc2', 'ising-mvse');


p = polyfit(T,M,3);
O=polyval(p,T);
figure()
plot(T,O)

function e = energy(a);
n = length(a);
sum = 0;
for i = 1:n
    for j = 1:n
        if j-1 < 1
            p = a(i,j) * a(i,n);
        else
            p = a(i,j) * a(i,j-1);
        end
        if j+1 > n
            q = a(i,j) * a(i,1);
        else
            q = a(i,j) * a(i,j+1);
        end
        if i-1 < 1
            r = a(i,j) * a(n,j);
        else
            r = a(i,j) * a(i-1,j);
        end
        if i+1 > n
            s = a(i,j) * a(1,j);
        else
            s = a(i,j) * a(i+1,j);
        end
        sum = sum + p + q + r + s;
    end
end
e = sum;
end
```

Representative Output:



10-by-10 Grid
Black = spin down
White = spin up