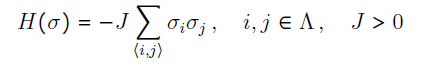
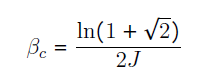
**Monte Carlo Simulations of the 2D Ising Model**

Student: Volodymyr Kuz, 300809

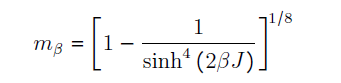
# Model Description

The 2D Ising model represents a system of binary spin variables σᵢ ∈ {−1, +1} arranged on a square lattice. Each spin interacts with its nearest neighbors. The energy (Hamiltonian) of a given spin configuration is defined as:  
  
   
  
where the summation runs over all nearest-neighbor pairs ⟨i,j⟩, and J > 0 is the coupling constant. In this project, we set J = 1 (ferromagnetic case).

In the thermodynamic limit, the 2D Ising model undergoes a phase transition at the inverse critical temperature:



For 𝛽 > 𝛽𝑐, the 2D Ising model exhibits a spontaneous magnetization



# Monte Carlo Method

Monte Carlo methods, or Monte Carlo experiments, are a broad class of computational algorithms that rely on repeated random sampling to obtain numerical results. The underlying concept is to use randomness to solve problems that might be deterministic in principle.

Monte Carlo methods are different, but tend to follow a particular pattern:

1) Define a domain of possible inputs

2) Generate inputs randomly from a probability distribution over the domain

3) Perform a deterministic computation on the inputs

4) Aggregate the results

# 3. Metropolis algorithm

Let Q be a symmetric transition matrix. Let be a distribution on with >0, for all .

We will use the notation for the states, i.e. the configurations of the spins.

The Algorithm:

1. **Inizialization**. We start choosing an initial state, that is an initial configuration for the spins .
2. **Trial move**. We propose a move as follows: we choose randomly according to the Markov chain Q a spin and flip it.
3. **Acceptation**. If we have flipped, say, the -th spin, we have

and .

Compute the acceptance probability as:

Then we generate a uniform random number and:

(a) if , accept the move, defining ;

(b) if , reject the move, defining .

1. **Iteration**. Then, we use as the new starting point, and go back to step2.

# 4. Initial Configurations

In our simulations set (in this case ) and

. Time is measured in number of sweeps .

Initial configurations. For all 𝑖 𝜖 Λ take:

1. = +1

2. = −1

3. = +1 with probability 1/2

Compute the following observables:

* the magnetization (per spin) as a function of time (for

2 < and 𝑇 = 2.5 > , and using the i.c. above);

* the energy as a function of time (for 2 < and

𝑇 = 2.5 > , and using the i.c. above);

* the mean magnetization 〈𝑚〉 as a function of the temperature T;
* the mean energy 〈𝑒〉 as a function of the temperature T;
* the microscopic configurations {} sampled at different times:

with =1,…,9 for 2 < and 𝑇 = 2.5 > .

# 5. Results Summary 1)Thermalization time

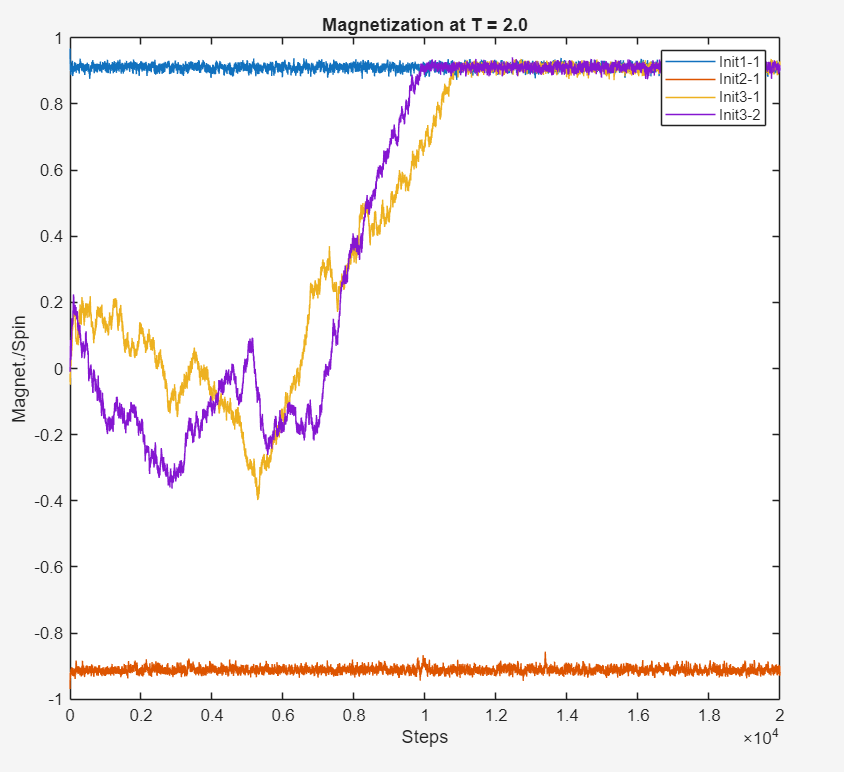


Figure 1: Magnetization per site as a function of time at temperature T = 2.

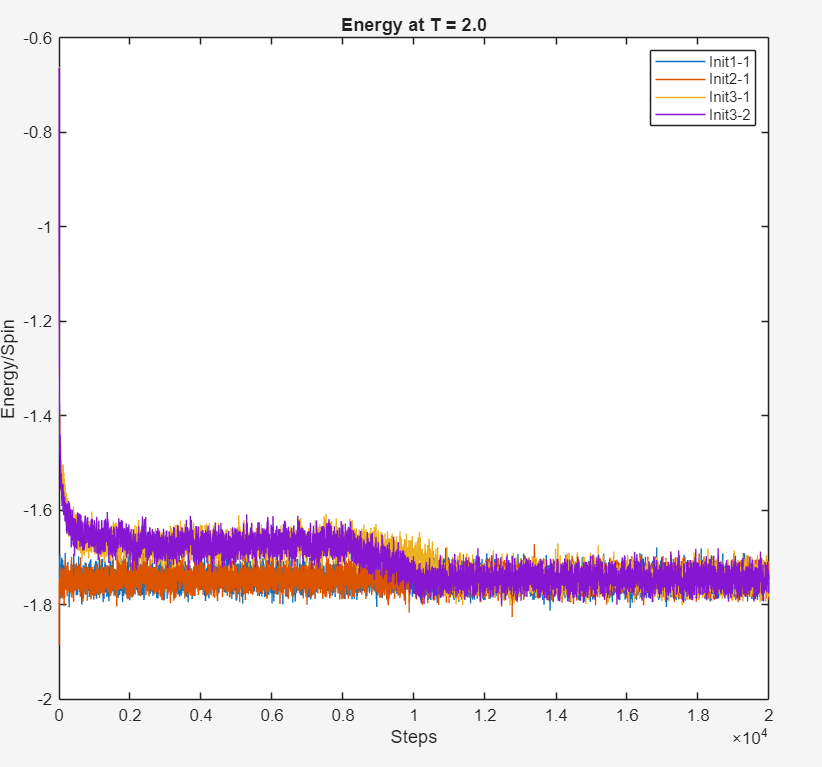


Figure 2: Energy per site at T = 2.

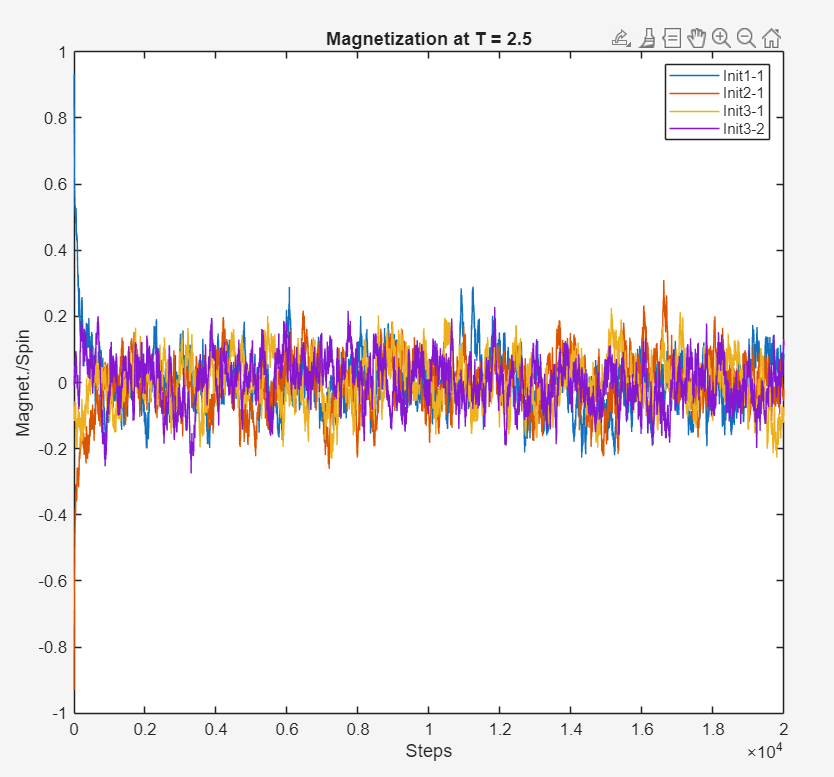


Figure 3: Magnetization per site as a function of time at temperature T = 2.5.

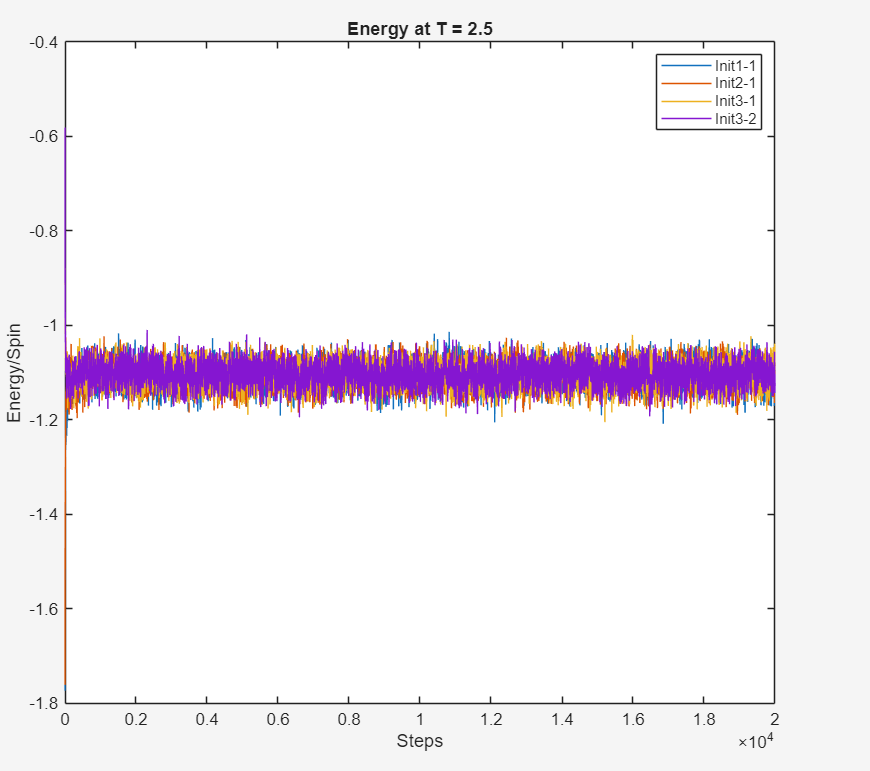


Figure 4: Energy per site at T = 2.5.

At 𝑇 = 2.0 (< 𝑇𝑐), the system tends to exhibit spontaneous magnetization. Configurations evolve toward full alignment, indicating an ordered phase. The magnetization remains near ±1, depending on the initial condition. The energy decreases rapidly and then stabilizes, indicating convergence toward thermal equilibrium in the ordered phase.

At 𝑇 = 2.5 (> 𝑇𝑐), the system displays disordered behavior. Magnetization decays toward zero, regardless of initial state. This suggests that thermal energy dominates and no long-range order is established. Results confirm that at high temperature, the system does **not exhibit long-range magnetic order.**

**2) Phase transition**

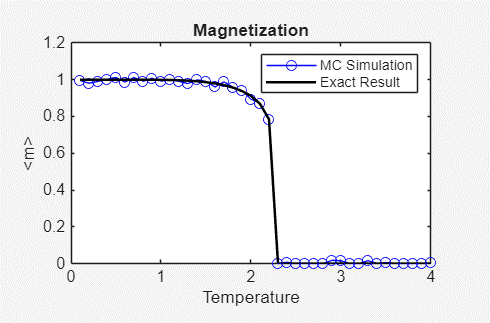


Figure 5: Mean magnetization 〈𝑚〉.

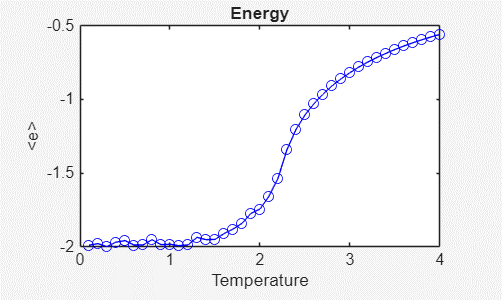


Figure 6: Mean energy 〈𝑒〉.

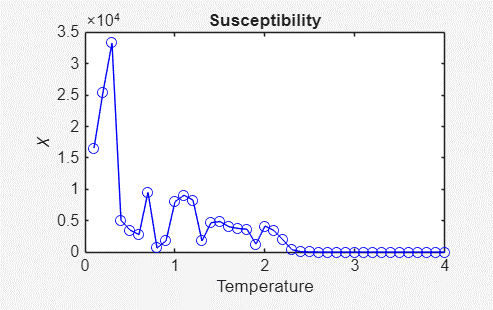


Figure 7: Magnetic Susceptibility X.

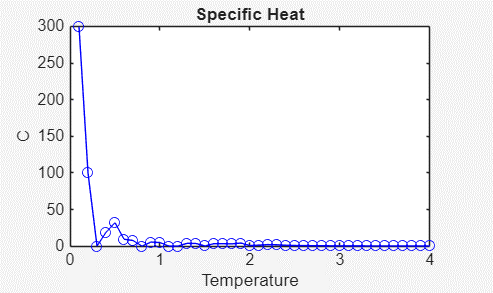


Figure 8: Specific heat c.

Plots of susceptibility and specific heat as functions of 𝑇 reveal peaks near 𝑇𝑐, confirming the presence of a second-order phase transition.

At **low temperatures**, the system exhibits spontaneous magnetization (⟨m⟩ ≠ 0), indicating an **ordered ferromagnetic phase.**  
As temperature increases and crosses 𝑇𝑐, ⟨m⟩ rapidly drops to zero, characterizing the **disordered paramagnetic phase.**

The energy increases smoothly with temperature, with a noticeable change in slope near 𝑇𝑐, reflecting changes in internal spin alignment.

Large fluctuations in magnetization near 𝑇𝑐 are responsible for this behavior.

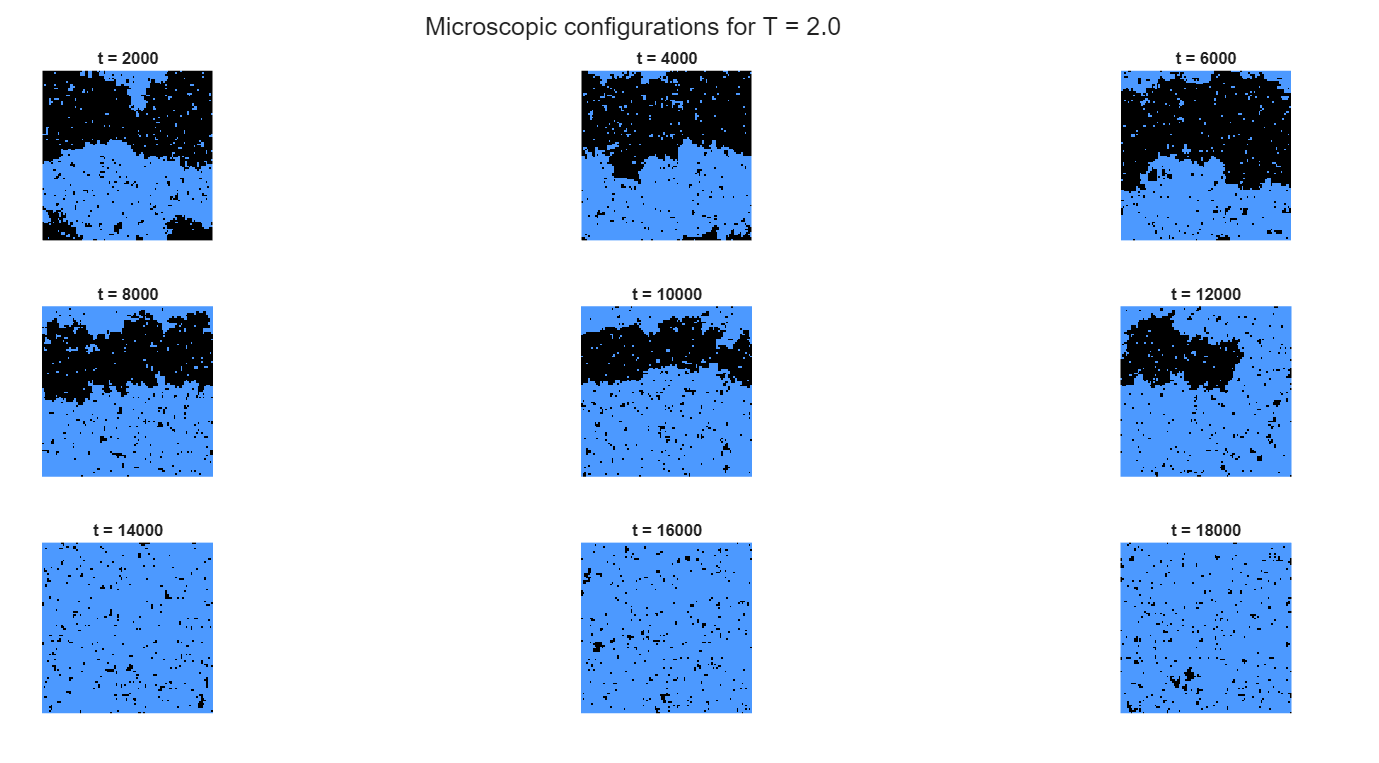
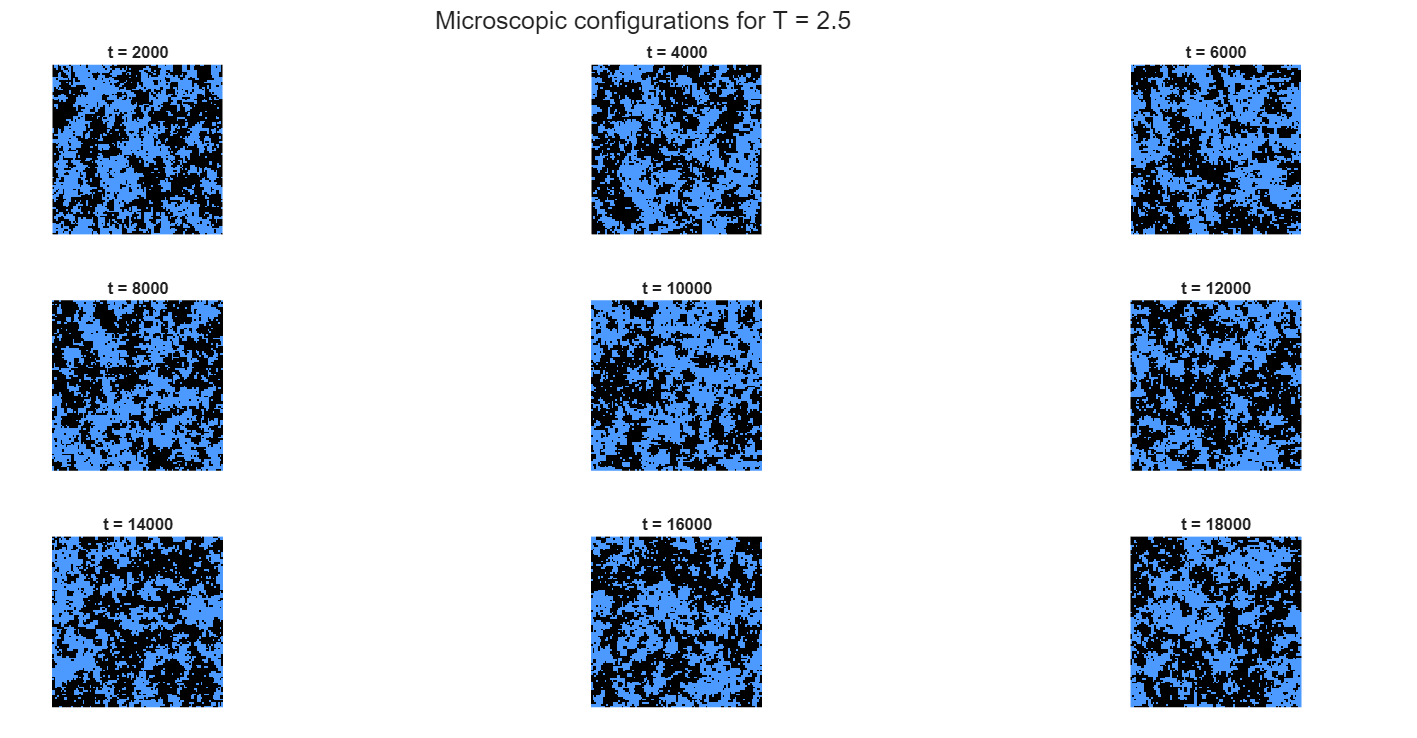
**3) Microscopic configurations**

Figure 9: Microscopic evolution of spin configurations {σᵢ} at temperature 2 < .

Each image corresponds to the lattice state at a specific Monte Carlo time step (from **t = 2000** to **t = 18000**, in steps of 2000).

* **Blue** pixels represent spins with σᵢ = +1
* **Black** pixels represent spins with σᵢ = −1

In this case, the system starts with a random configuration and gradually evolves toward a fully magnetized phase. Large **aligned domains emerge** and grow over time, eventually dominating the lattice. This behavior reflects the **spontaneous symmetry breaking** characteristic of the **ferromagnetic phase below the critical temperature.**

Figure 10: Microscopic spin configurations for 𝑇 = 2.5 >

At this temperature, which is above the critical point **≈ 2.269**, the system remains **disordered** throughout the simulation. No large-scale domains emerge, and the spin distribution fluctuates chaotically around zero magnetization, reflecting the **paramagnetic phase.**

# 7. Conclusions

The simulation of the 2D Ising model using the Metropolis Monte Carlo algorithm effectively reproduced the expected behavior of ferromagnetic systems near the critical temperature .

We observed:

* **Spontaneous magnetization** and domain formation in the ordered phase for **T <** .
* A **disordered, paramagnetic phase** for **T >** , where the magnetization fluctuates around zero
* Clear **peaks in magnetic susceptibility and specific heat** near , confirming the presence of a **second-order phase transition**

In addition to time series analysis, we visualized **microscopic spin configurations** at multiple Monte Carlo steps. These snapshots illustrated the growth of aligned domains below , and the persistent disorder above , offering a compelling spatial representation of the transition dynamics.

The model was implemented in MATLAB with the following settings:  
- Lattice size: 100 × 100  
- Total sweeps: 20000  
- Temperature range: 0.1 to 4.0  
- Random initialization with fixed seed for reproducibility

Key MATLAB functions:

* initSpinConfiguration — initialize spin lattice
* applyMetropolis — perform Metropolis updates over the lattice
* calculateEnergy and calculateMagnetization — compute physical observables
* Microscopic snapshots and time series plotting were used for comprehensive analysis

Below is the code used for the implementation:

|  |
| --- |
| % 2D Ising Model Simulation using Metropolis Monte Carlo Algorithm  % Author: Volodymyr Kuz  % Course: Kinetic Theory and Stochastic Simulations  %% Simulation Parameters  latticeSize = 100;  totalSpins = latticeSize^2;  sweepCount = 20000;  temperatureSet = [2.0, 2.5];  temperatureGrid = linspace(0.1, 4.0, 40);  rng(1); % Fix seed for reproducibility  %% Utility Functions  function spinMatrix = initSpinConfiguration(sizeL, mode)  if mode == 1  spinMatrix = ones(sizeL);  elseif mode == 2  spinMatrix = -ones(sizeL);  else  spinMatrix = sign(rand(sizeL) - 0.5);  end  end  function spinMatrix = applyMetropolis(spinMatrix, invTemp)  L = size(spinMatrix,1);  for step = 1:L^2  x = randi(L); y = randi(L);  currentSpin = spinMatrix(x,y);  neighbours = spinMatrix(mod(x,L)+1,y) + spinMatrix(mod(x-2,L)+1,y) + ...  spinMatrix(x,mod(y,L)+1) + spinMatrix(x,mod(y-2,L)+1);  deltaE = 2 \* currentSpin \* neighbours;  if deltaE < 0 || rand < exp(-invTemp \* deltaE)  spinMatrix(x,y) = -currentSpin;  end  end  end  function eVal = calculateEnergy(spinMatrix)  L = size(spinMatrix,1); eVal = 0;  for row = 1:L  for col = 1:L  s = spinMatrix(row,col);  nbs = spinMatrix(mod(row,L)+1,col) + spinMatrix(row,mod(col,L)+1);  eVal = eVal - s \* nbs;  end  end  end  function mVal = calculateMagnetization(spinMatrix)  mVal = sum(spinMatrix(:));  end  %% Time Evolution at Fixed Temperatures  for T = temperatureSet  beta = 1/T;  figure('Name',sprintf('Temporal Evolution at T = %.1f',T));  tiledlayout(1,2);  for startCase = 1:3  repeatNum = 2 \* (startCase == 3) + 1 \* (startCase < 3);  for instance = 1:repeatNum  config = initSpinConfiguration(latticeSize, startCase);  magnetHistory = zeros(1, sweepCount);  energyHistory = zeros(1, sweepCount);  for t = 1:sweepCount  config = applyMetropolis(config, beta);  magnetHistory(t) = calculateMagnetization(config)/totalSpins;  energyHistory(t) = calculateEnergy(config)/totalSpins;  end  nexttile(1)  plot(magnetHistory, 'DisplayName', sprintf('Init%d-%d',startCase,instance)); hold on  title(sprintf('Magnetization at T = %.1f', T)); xlabel('Steps'); ylabel('Magnet./Spin')  nexttile(2)  plot(energyHistory, 'DisplayName', sprintf('Init%d-%d',startCase,instance)); hold on  title(sprintf('Energy at T = %.1f', T)); xlabel('Steps'); ylabel('Energy/Spin')  end  end  nexttile(1); legend; nexttile(2); legend;  end  %% Thermal Averages Across Temperature Grid  avgMag = zeros(size(temperatureGrid));  avgEnergy = zeros(size(temperatureGrid));  chiVals = zeros(size(temperatureGrid));  heatVals = zeros(size(temperatureGrid));  for idx = 1:length(temperatureGrid)  T = temperatureGrid(idx); beta = 1/T;  mSamples = []; eSamples = [];  for trial = 1:8  config = initSpinConfiguration(latticeSize, 3);  for s = 1:sweepCount  config = applyMetropolis(config, beta);  if s > sweepCount/2  m = calculateMagnetization(config)/totalSpins;  e = calculateEnergy(config)/totalSpins;  mSamples(end+1) = m;  eSamples(end+1) = e;  end  end  end  avgMag(idx) = movmean(abs(mean(mSamples)), 5);  avgEnergy(idx) = mean(eSamples);  chiVals(idx) = beta\*totalSpins\*(mean(mSamples.^2) - mean(mSamples)^2);  heatVals(idx) = beta^2\*totalSpins\*(mean(eSamples.^2) - mean(eSamples)^2);  end  % exact result for mean magnetization below Tc  Tc = 2 / log(1 + sqrt(2)); % ≈ 2.269  exactM = zeros(size(temperatureGrid));  for i = 1:length(temperatureGrid)  T = temperatureGrid(i);  if T < Tc  exactM(i) = (1 - sinh(2/T)^(-4))^(1/8);  else  exactM(i) = 0;  end  end  figure('Name','Analytical vs Simulated Results');  subplot(2,2,1);  plot(temperatureGrid, avgMag, 'bo-', 'DisplayName', 'MC Simulation'); hold on;  plot(temperatureGrid, exactM, 'k-', 'LineWidth', 1.5, 'DisplayName', 'Exact Result');  xlabel('Temperature'); ylabel('<m>'); title('Magnetization');  legend;  subplot(2,2,2);  plot(temperatureGrid, avgEnergy, 'bo-');  xlabel('Temperature'); ylabel('<e>'); title('Energy');  subplot(2,2,3);  plot(temperatureGrid, chiVals, 'bo-');  xlabel('Temperature'); ylabel('\chi'); title('Susceptibility');  subplot(2,2,4);  plot(temperatureGrid, heatVals, 'bo-');  xlabel('Temperature'); ylabel('C'); title('Specific Heat'); |

Microscopic spin configurations

|  |
| --- |
| % Generate microscopic spin configurations at different times  L = 100;  N = L^2;  sweepCount = 20000;  saveSteps = 2000:2000:18000; % 9 snapshots  temps = [2.0, 2.5];  % Color map: -1 -> black, +1 -> blue  customCMap = [0 0 0; 0.3 0.6 1];  for Tidx = 1:2  T = temps(Tidx);  beta = 1/T;  config = sign(rand(L)-0.5); % random init  snapList = {};  stepIndex = 1;  for sweep = 1:sweepCount  config = metropolisStep(config, beta);  if ismember(sweep, saveSteps)  snapList{stepIndex} = config;  stepIndex = stepIndex + 1;  end  end  figure('Name', sprintf('Microscopic Configurations at T = %.1f', T), 'Color', 'w');  for i = 1:9  subplot(3,3,i);  imagesc(snapList{i});  axis square off;  colormap(customCMap);  caxis([-1 1]);  title(sprintf('t = %d', saveSteps(i)));  end  sgtitle(sprintf('Microscopic configurations for T = %.1f', T));  end  % --- Metropolis step function ---  function config = metropolisStep(config, beta)  L = size(config,1);  for k = 1:L^2  i = randi(L); j = randi(L);  s = config(i,j);  nb = config(mod(i,L)+1,j) + config(mod(i-2,L)+1,j) + ...  config(i,mod(j,L)+1) + config(i,mod(j-2,L)+1);  dE = 2 \* s \* nb;  if dE < 0 || rand < exp(-beta \* dE)  config(i,j) = -s;  end  end  end |