Computational Physics Project

Many Body Physics

Submitted by
Ramit Goyal
Roll No.: 25B1049

Mentor
Arnav Jain

Maths and Physics Club, IIT Bombay

December 31, 2025

Contents

1 Part A: Percolation Theory

1.1 Introduction

Percolation theory provides a probabilistic model for studying connected clusters in a random graph. It is fundamental to understanding phase transitions, critical phenomena, and transport properties in disordered systems.

In this project, we implement a numerical simulation of **Site Percolation** on a 2D square lattice of size $L \times L$. Sites are randomly occupied (filled black) with probability p. The primary objective is to investigate the phase transition that occurs at a critical probability p_c, where a cluster emerges, connecting opposite boundaries of the lattice.

1.2 Computational Methods

The simulation is built upon four primary components: Grid Initialization, Cluster Identification (Hoshen-Kopelman), Percolation Detection, and Monte Carlo Analysis.

1.2.1 1. Grid Initialization

We generate an $L \times L$ grid where each site (i, j) is occupied (1) with probability p and empty (0) with probability $1 - p$. This is achieved by generating a matrix of uniform random numbers $r \in [0, 1)$ and applying a threshold.

Algorithm 1 Grid Initialization

```
function GenerateGrid(n, p)
    R ← random n × n matrix with values in [0, 1)
    grid ← (R < p)                    ▷ Boolean matrix: True if occupied
    return grid
end function
```

1.2.2 2. Cluster Identification: Hoshen-Kopelman Algorithm

To identify distinct clusters, we implement the **Hoshen-Kopelman algorithm** enhanced with a **Union-Find** data structure. This algorithm scans the grid once to assign provisional labels and resolve equivalences between connected neighbors.

Connectivity Rule: Sites are connected if they are nearest neighbors (Up, Down, Left, Right). In the first pass (Top-to-Bottom, Left-to-Right), we only need to check the Left and Up neighbors.

Algorithm 2 Hoshen-Kopelman with Union-Find (User Implementation)

```
function HoshenKopelman(grid)
    n ← length of grid
    labels ← n × n zeros matrix
    label_counter ← 0
    parent ← dictionary for Union-Find
    Helper: Find(x) returns root of x (with path compression)
    Helper: Union(x, y) links root of y to root of x
    for i ← 0 to n − 1 do
        for j ← 0 to n − 1 do
            if grid[i][j] is Occupied then
                neighbors ← []
                if i > 0 and labels[i − 1][j] > 0 then neighbors.append(labels[i − 1][j])
                end if
                if j > 0 and labels[i][j − 1] > 0 then neighbors.append(labels[i][j − 1])
                end if
                if neighbors is empty then
                    label_counter ← label_counter + 1
                    labels[i][j] ← label_counter
                    parent[label_counter] ← label_counter
                else
                    min_label ← min(neighbors)
                    labels[i][j] ← min_label
                    for neighbor in neighbors do
                        Union(min_label, neighbor)
                    end for
                end if
            end if
        end for
    end for
                                                    ▷ Second Pass: Flatten labels
    for i ← 0 to n − 1 do
        for j ← 0 to n − 1 do
            if labels[i][j] > 0 then
                labels[i][j] ← Find(labels[i][j])
            end if
        end for
    end for
    return labels
end function
```

1.2.3 3. Percolation Detection

We define the system as percolating if there is a cluster label present in the top row that is also present in the bottom row.

Algorithm 3 Check Percolation

```
function CheckPercolation(labels)
    n ← length of labels
    top_set ← unique labels in row 0 (excluding 0)
    bottom_set ← unique labels in row n − 1 (excluding 0)
    common ← top_set ∩ bottom_set
    if common is not empty then
        return True
    else
        return False
    end if
end function
```

1.3 Simulation Parameters & Analysis

We performed Monte Carlo simulations to calculate statistical averages.

- **Lattice Sizes (L):** 8, 16, 32.
- **Probability (p):** Linearly spaced values from 0 to 1.
- **Trials:** Averaged over N realizations per (L, p) pair (e.g., 100-300 trials).

1.3.1 Calculated Quantities

1. Weighted Average Cluster Size (S): This measures the expected size of a cluster chosen by picking a random site.

$$S(p) = \frac{\sum_s s^2 n_s}{\sum_s s n_s} \tag{1}$$

where n_s is the number of clusters of size s.

2. Percolation Strength (P): The order parameter of the phase transition. It is the fraction of sites belonging to the percolating cluster.

$$P = \frac{\text{Size of percolating cluster}}{L^2} \tag{2}$$

If no percolation occurs, P = 0.

3. Percolation Rate / Probability (Π): The probability that a random lattice at occupation p contains a spanning cluster.

$$\Pi(p) = \frac{\text{Count of Percolating Trials}}{\text{Total Trials}} \tag{3}$$

1.4 Results and Observations

1.4.1 Visualizing the Lattice

Before analyzing the statistics, we visualized the grid to understand the qualitative behavior of clusters. Figure 1 shows the lattice state for a specific configuration.

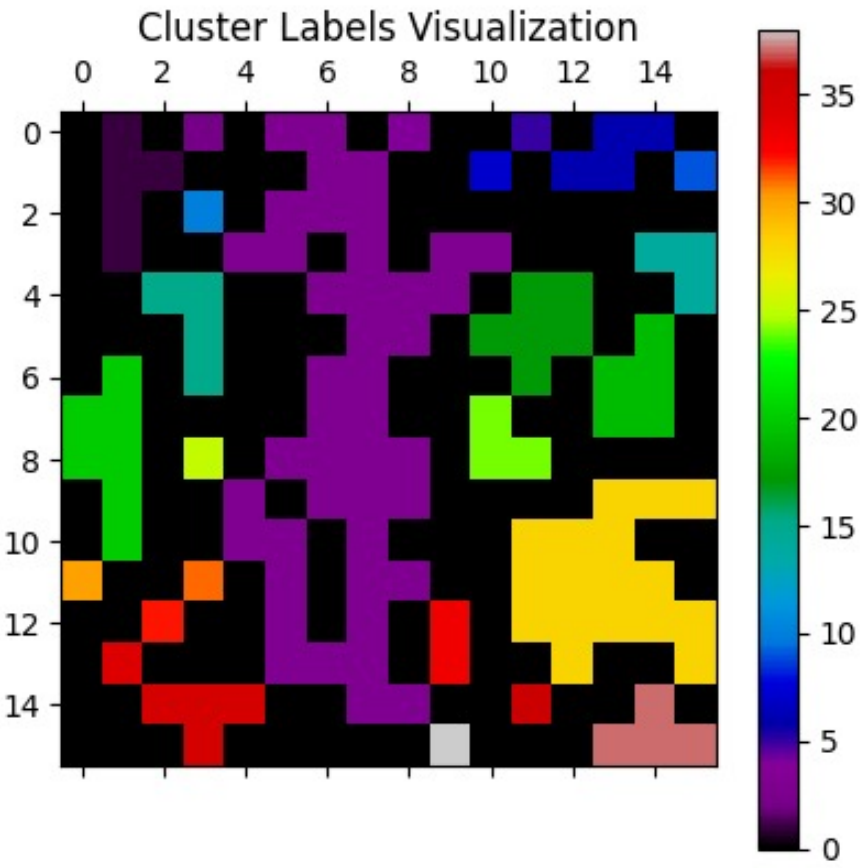

Figure 1: Visualization of clusters on a 2D square lattice for **n=16** and **p=0.5**. Distinct colors represent different connected components identified by the Hoshen-Kopelman algorithm.

1.4.2 Phase Transition Analysis: Site Percolation

We analyzed the system behavior for lattice sizes $L \in \{8, 16, 32\}$. The following plots illustrate the phase transition behavior.

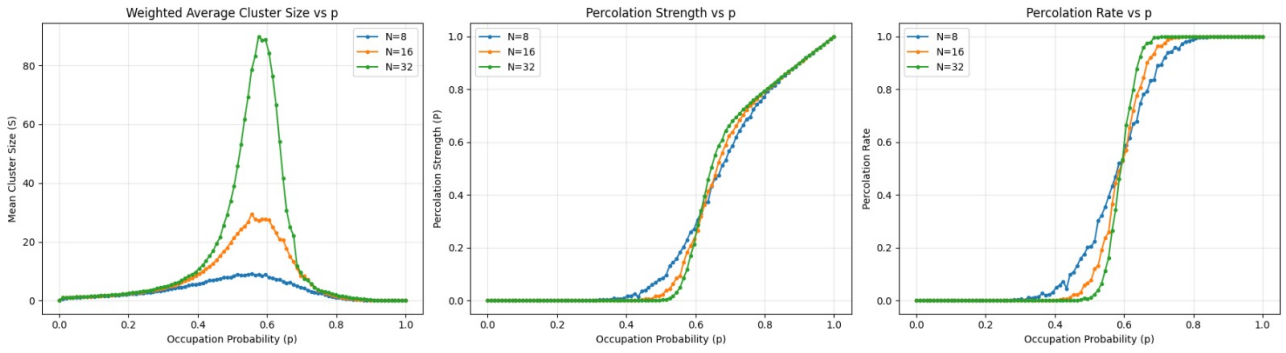

Figure 2: Site Percolation Results: (Left) Percolation Strength P_∞, (Center) Percolation Probability $\Pi(p)$, and (Right) Average Cluster Size S vs. Probability p.

From Figure 2, we observe:

- **Percolation Probability**: Exhibits a sigmoid (S-shape) transition from 0 to 1. As L increases, the transition becomes sharper. All curves intersect near $p \approx 0.59$.

- **Percolation Strength** (P_∞): For $p < p_c$, the strength is effectively zero. At $p > p_c$, there is a sharp increase.
- **Weighted Average Cluster Size** (S): Shows a distinct peak near $p_c \approx 0.59$. The height of the peak increases dramatically with system size L.

1.4.3 Comparison with Bond Percolation

In addition to site percolation, we simulated **Bond Percolation** on the same square lattice.

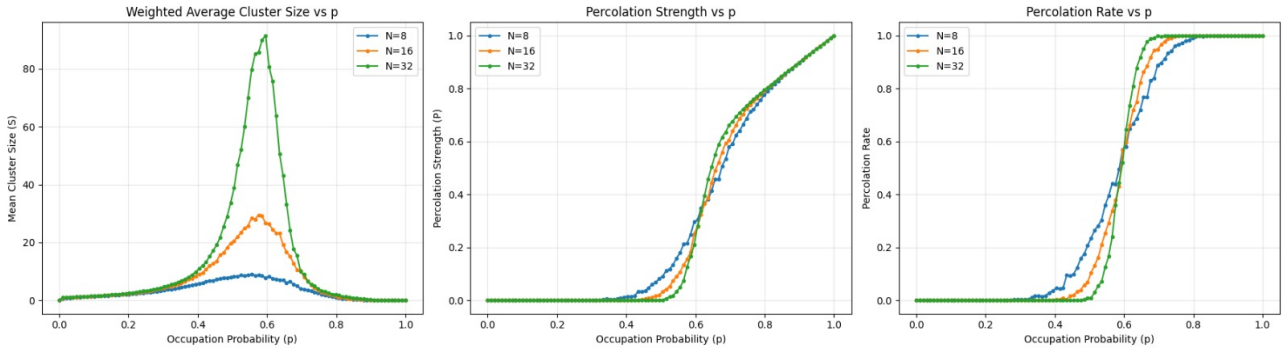

Figure 3: Bond Percolation Results for $L = 8, 16, 32$. The transition occurs distinctly earlier than in site percolation.

As seen in Figure 3, the critical threshold shifts:

- Site Percolation p_c: ≈ 0.5927
- Bond Percolation p_c: ≈ 0.5000 (Matches theoretical $p_c = 1/2$)

1.4.4 Theoretical Comparison: The Bethe Lattice

For a Bethe lattice with coordination number z, the critical probability is given by $p_c = \frac{1}{z-1}$.

2 Part B: Ising Model

2.1 Introduction

The Ising Model is a fundamental model in statistical mechanics used to study phase transitions in ferromagnetic materials. The system consists of discrete spins $\sigma_i \in \{+1, -1\}$ arranged on a lattice, interacting via nearest-neighbor coupling.

The Hamiltonian of the system is given by:

$$H(\sigma) = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j \tag{4}$$

where J is the coupling constant ($J > 0$ favors alignment/ferromagnetism) and the sum runs over adjacent pairs. In this project, we simulate the 2D Ising model to investigate the spontaneous magnetization and the second-order phase transition occurring at the critical temperature T_c.

2.2 Computational Methods

2.2.1 1. Grid Initialization

The simulation begins by generating an $L \times L$ lattice.

- **Disordered Start ($T = \infty$):** Spins are assigned $+1$ or -1 randomly.
- **Ordered Start ($T = 0$):** All spins are aligned (e.g., all $+1$).

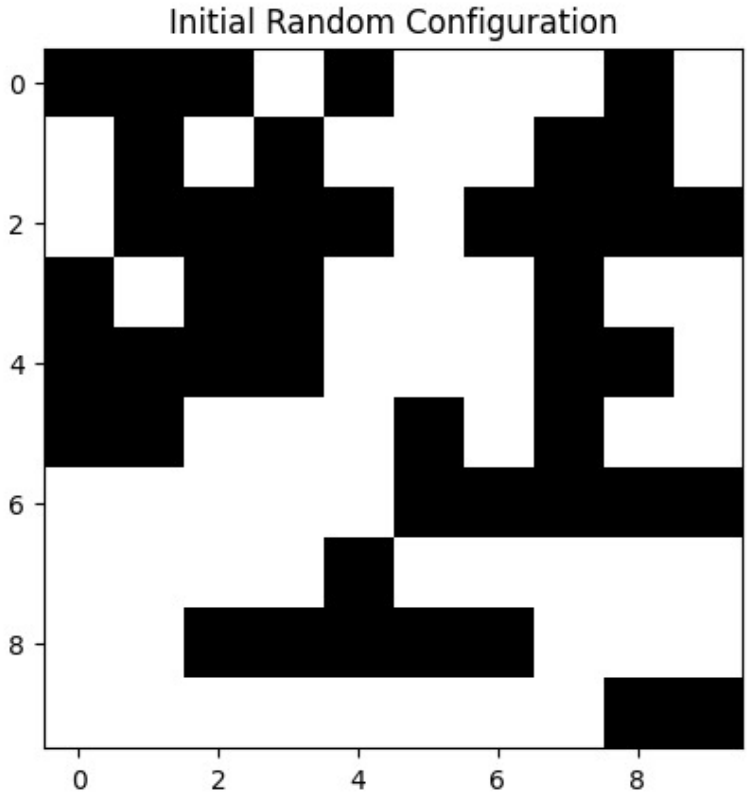

Figure 4: Visualization of the $L \times L$ grid with random spin initialization.

2.2.2 2. Metropolis Algorithm (Single Spin Flip)

To simulate the system at a finite temperature T, we use the single spin flip algorithm:

1. Pick a lattice site randomly.
2. Calculate the change in energy ΔE required to flip the spin.
3. Accept the flip if $\Delta E < 0$, or with probability $e^{-\Delta E/k_B T}$ if $\Delta E > 0$.

This local update method is simple but suffers from high autocorrelation times near criticality.

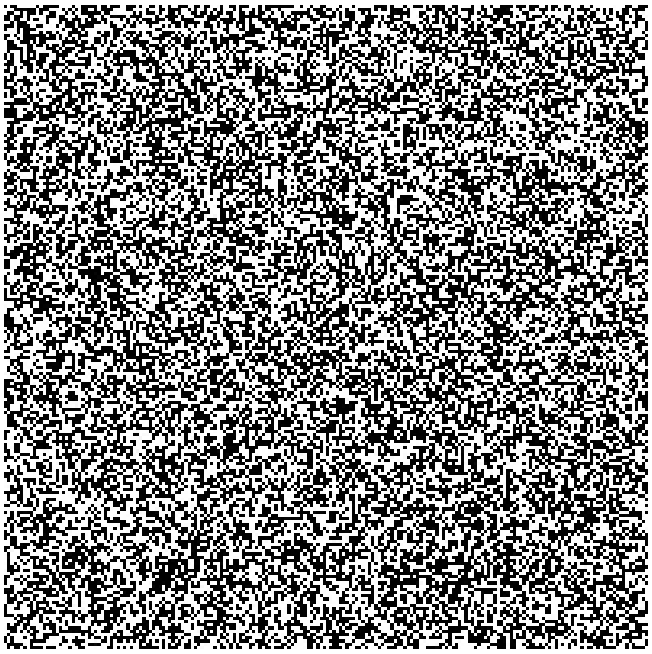

Figure 5: Visualization of 200*200 grid with random spin initialization.

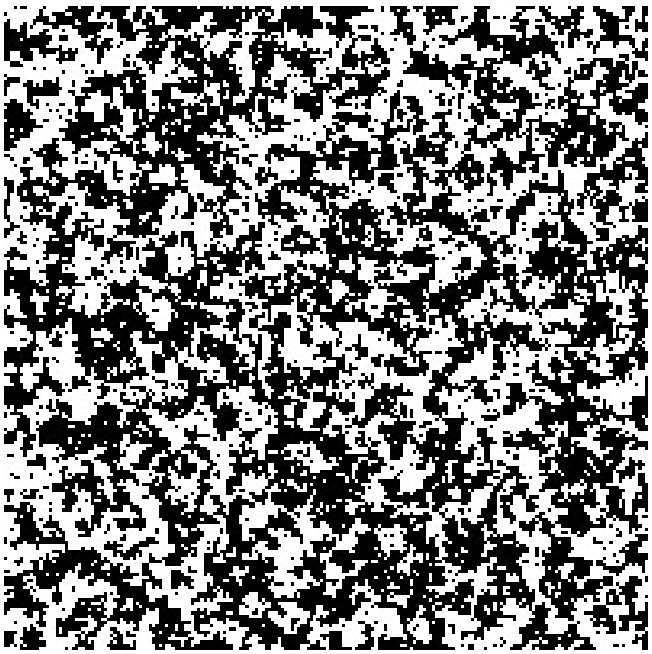

Figure 6: Visualization of 200*200 grid with one ising step.

2.2.3 3. Wolff Algorithm (Cluster Flip)

To mitigate "Critical Slowing Down," we implemented the Wolff Cluster algorithm:

1. Select a random seed spin.
2. Grow a cluster by adding neighbors with probability $P = 1 - e^{-2J/k_BT}$.
3. Flip the entire cluster simultaneously.

This global update method significantly reduces equilibration time near T_c.

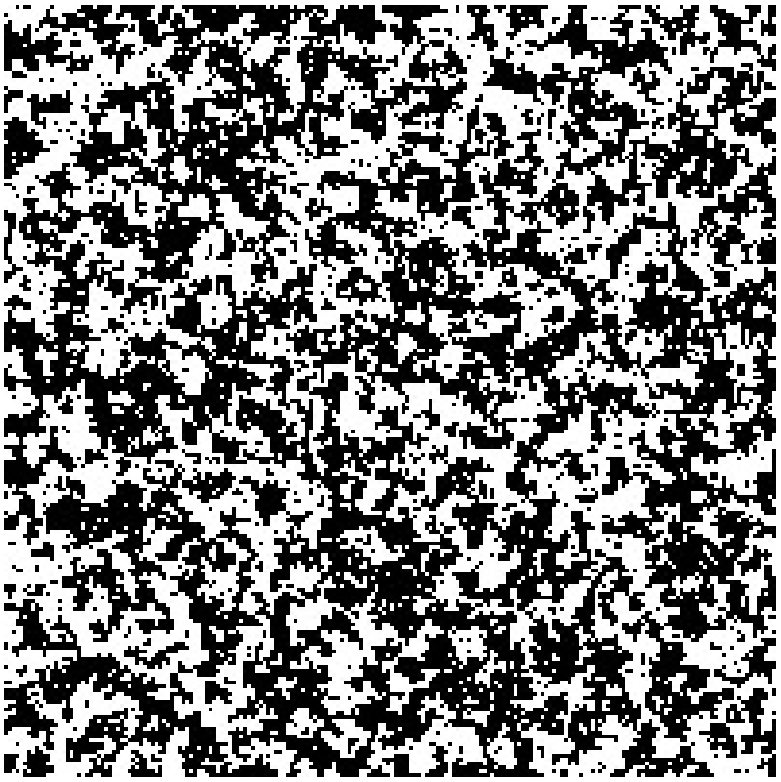

Figure 7: Visualization of 200*200 grid with one ising step via Wolff algorithm.

2.3 Simulation Parameters & Analysis

2.3.1 Calculated Quantities

We monitor the following macroscopic observables:

- **Magnetization (M):** $\frac{1}{N}\sum \sigma_i$
- **Magnetic Susceptibility (χ):** Fluctuation in M.
- **Specific Heat (C_v):** Fluctuation in Energy.

2.3.2 Binning Analysis

To estimate errors correctly from correlated Monte Carlo data, we utilized the Binning Analysis method. By averaging measurements over blocks of time, we calculate the **Autocorrelation Time (τ)**, which quantifies the efficiency of our sampling algorithms.

2.4 Results and Observations

2.4.1 Visualizing Equilibration

We observed the system evolving from a random noise state to stable ferromagnetic domains as the simulation progressed.

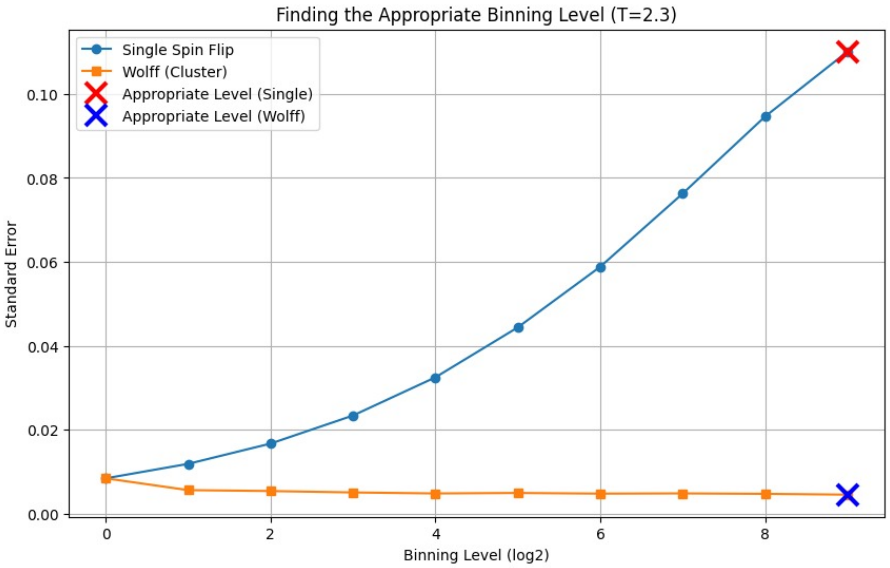

Figure 8: Snapshots of the grid showing domain formation over time.

2.4.2 Metropolis vs. Wolff Efficiency

The Wolff algorithm demonstrated a much shorter autocorrelation time compared to single spin flips, particularly near $T_c \approx 2.27$.

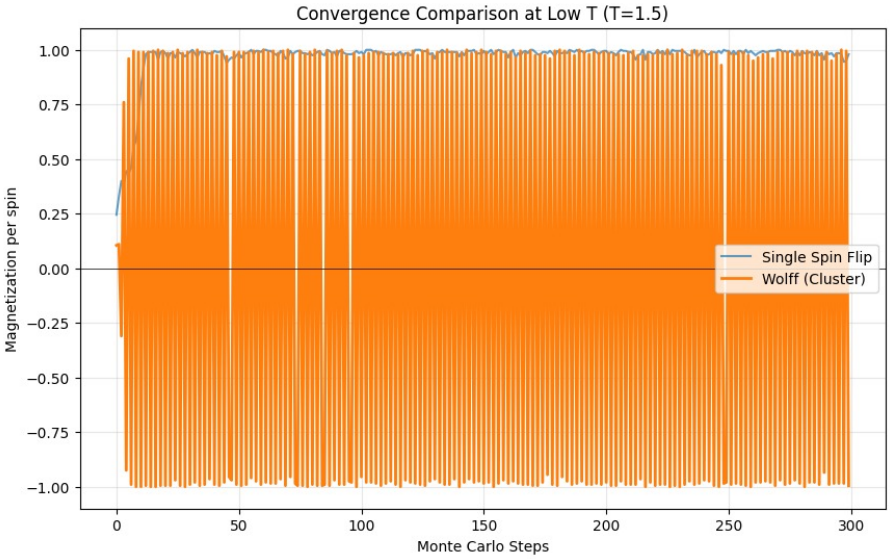

Figure 9: Comparison of autocorrelation times for local vs global updates.

2.4.3 Phase Transition: Magnetization vs Temperature

Plotting the average magnetization $\langle |M| \rangle$ against temperature reveals the phase transition. Below T_c, spontaneous magnetization is non-zero (ordered phase). Above T_c, it vanishes (disordered phase).

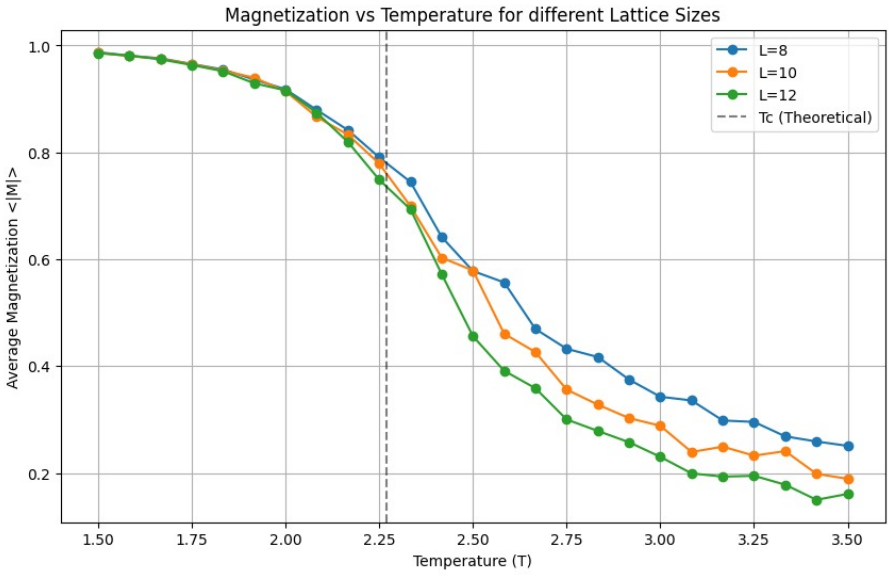

Figure 10: The order parameter M drops sharply to zero at the critical temperature.

2.4.4 Phase Transition: Autocorrelation Time vs Temperature

We analyzed the decay of the autocorrelation function $C(t)$ over Monte Carlo steps. The Metropolis algorithm shows slow decay near criticality due to local updates, whereas the Wolff algorithm exhibits rapid exponential decay, demonstrating its superior efficiency in decorrelating the system.

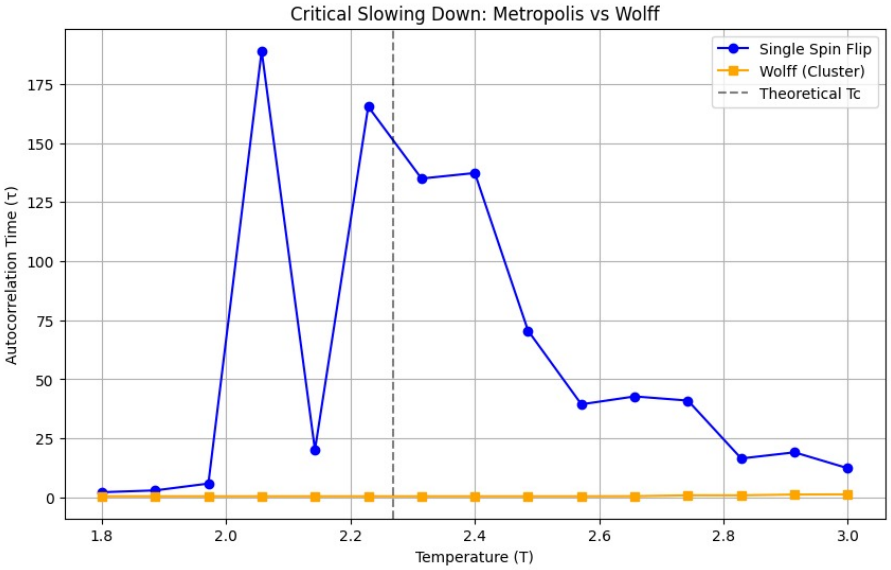

Figure 11: Decay of the Autocorrelation Function $C(t)$ with Monte Carlo steps. The Wolff algorithm (Cluster Flip) decorrelates significantly faster than the Metropolis algorithm (Single Spin Flip).

2.4.5 The XY Model and Topological Defects

In the final phase of our project, we simulated the **2D XY Model**, where spins possess continuous $O(2)$ rotational symmetry ($\mathbf{S}_i = (\cos\theta_i, \sin\theta_i)$). According to the Mermin-Wagner theorem, continuous symmetries cannot be spontaneously broken in 2D at finite temperatures. However, the system undergoes a topological phase transition known as the **Kosterlitz-Thouless (KT) transition**. Below T_{KT}, the system exhibits quasi-long-range order with bound vortex-antivortex pairs. Above T_{KT}, these pairs unbind, leading to a disordered phase. We visualized the local spin orientations to observe these topological defects (vortices) directly.

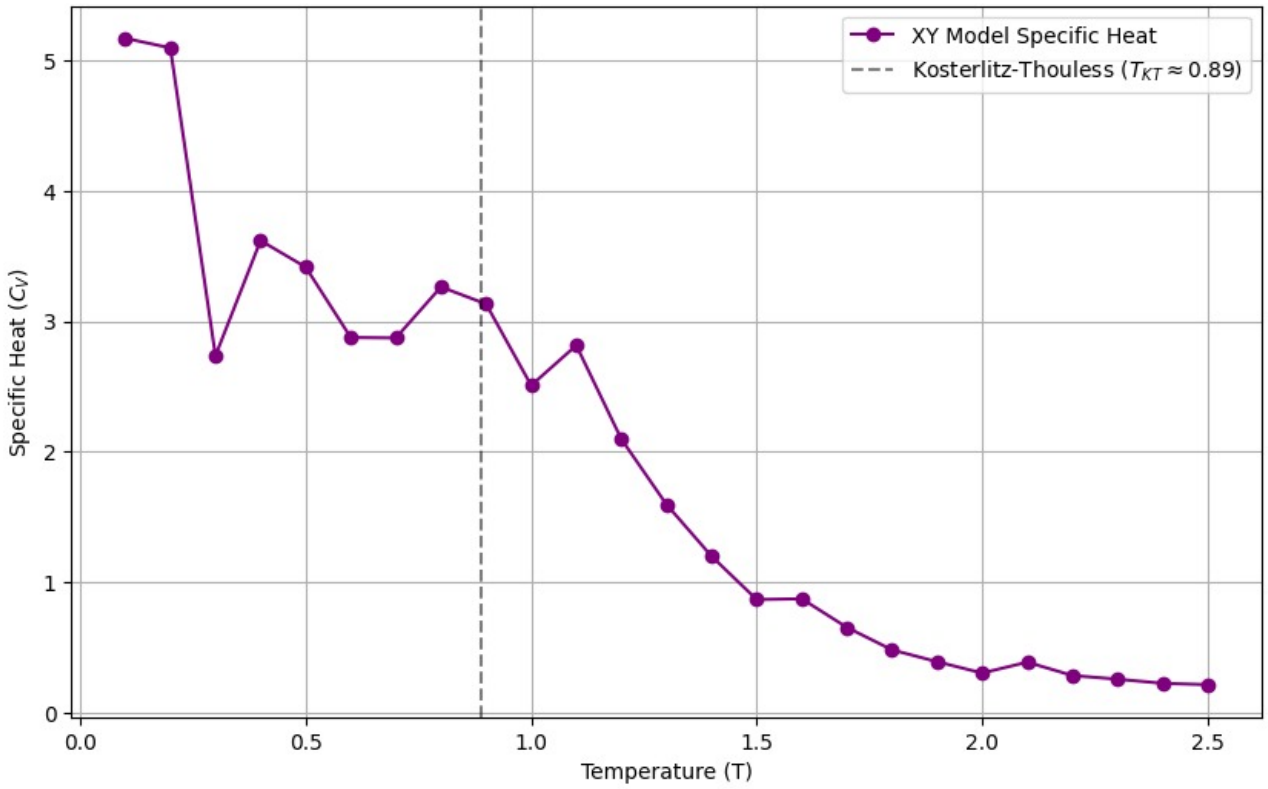

Figure 12: Snapshot of the 2D XY Model spins. The continuous rotational symmetry allows for the formation of vortices (swirling structures), which drive the Kosterlitz-Thouless transition.

2.4.6 Principal Component Analysis (PCA)

To visualize the phase separation in high-dimensional configuration space, we applied PCA to the lattice configurations. The projection onto the first two principal components clearly separates the low-temperature (ordered) states from high-temperature (disordered) states.

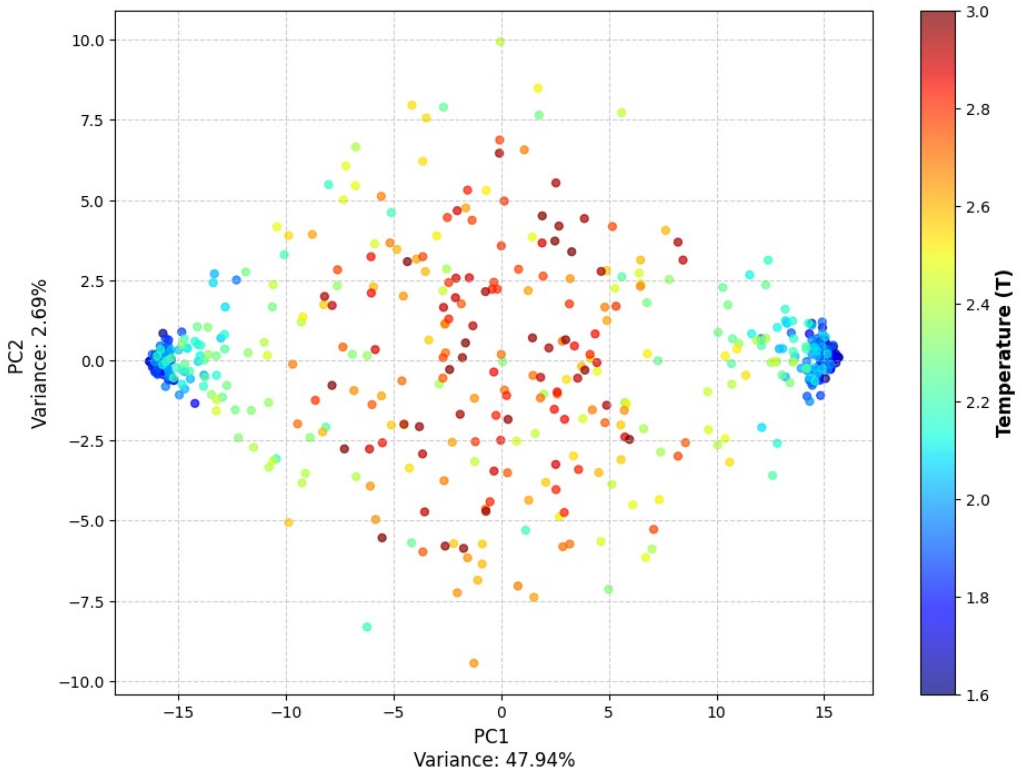

Figure 13: PCA projection of system configurations colored by temperature. Distinct clusters correspond to the two ordered ground states (all up/all down) and the disordered cloud.

2.5 Critical Temperature Estimation

Based on the divergence of susceptibility and specific heat peaks, we estimate the critical temperature to be:

$$T_c \approx 2.27$$

This is consistent with the real known solution for the 2D square lattice.

3 Overall Conclusion

In this simulation project, we explored critical phenomena in two distinct statistical systems:

1. **Percolation Theory (Part A):** We identified geometric phase transitions, finding $p_c \approx 0.59$ for site percolation and $p_c \approx 0.50$ for bond percolation.
2. **Ising Model (Part B):** We implemented simulations (Single Spin Flip and Wolff) to observe thermodynamic phase transitions, verifying the critical temperature $T_c \approx 2.27$.

Both parts highlight the importance of finite size scaling, efficient algorithms (Union-Find, Wolff), and the universality of critical behavior in many body physics.