PHYSICS CLUB
OCTOBER STEM SCHOOL

COMPASS

Comparative Analysis: Metropolis, Heat Bath, Wolff Cluster, and Swendsen-Wang Algorithms for a 2D Ising Model

Moaz Taha, Bishoy Hanna

February 8, 2026

STEM October Physics Club, COMPASS Computational Physics Program

Abstract

The two-dimensional Ising model is a fundamental tool for understanding phase transitions in statistical mechanics. However, simulations become computationally expensive near the critical temperature (T_C) due to "critical slowing down" and at low temperatures. This paper presents a comparative analysis of four Monte Carlo algorithms: Metropolis, Heat Bath, Wolff Cluster, and Swendsen–Wang applied to a 2D square lattice. The efficiency of these algorithms is investigated by calculating the integrated autocorrelation time (τ) and CPU time across a different range of temperatures. Results confirmed that the phase transition is at $T_C = 2.269\, J/k_B$ and demonstrate that while local update algorithms: Metropolis and Heat Bath become insufficient near criticality, the cluster-based algorithms: Wolff and Swendsen–Wang efficiently eliminate this effect. However, at very low temperatures, cluster algorithms become inefficient as clusters approach the system size, making local update methods relatively more suitable. These findings highlight the necessity of selecting the appropriate algorithm based on the thermal regime.

Keywords: N-body Simulation; Barnes-Hut; Particle-Mesh; Time Complexity Optimization

1 Introduction

Predicting the behavior and evolution of some physical deterministic systems, such as N-Body celestial systems and double-pendulum systems, or stochastic (probabilistic) systems, including Ising models, is analytically difficult or impossible. Therefore, computational solutions and simulations are effectively used to approximate the results of real-life systems by converting the continuous mathematics into discrete variables.

Ising models are mathematical models used in multiple research areas. In physics, Ising models focus on ferromagnetism, phase transitions, binary alloys, etc. In this project, an Ising model is used to simulate the behavior of particles and their interactions in a lattice of a ferromagnet. A lattice is a grid that represents the material's atomic geometry. The most popular lattice, and the one used in this project, is the 2D lattice. In the 2D lattice, atoms are arranged in rows and columns, each with a spin (σ) of either $+1$ (up) or -1 (down), such that an individual atom interacts with its four neighbors: one on top of it, under it, on its right, and on its left. Interactions between atoms naturally tend to lower the energy of the system by aligning them, either in the upward or downward direction; such interactions are energetically and statistically favorable due to their higher probability.

However, sampling the behavior of the atoms in a ferromagnetic material in its equilibrium state near the critical temperature (T_C) is computationally expensive because of the critical slowing down. For this, the paper discusses and compares the performance of four Monte Carlo algorithms: the Metropolis Algorithm, the Heat Bath Algorithm, the Wolff Algorithm, and the Swendsen–Wang Algorithm, which were run on Ising model simulations of various pairs of lattice sizes (L) and temperature (T). It analyzes the computational efficiency and the accuracy of physics observables by measuring:

- CPU time per independent sample vs. Temperature (T).
- Integrated autocorrelation time (τ) vs. Temperature (T).
- Average Absolute Magnetization $\langle|m|\rangle$, Average Energy $\langle e\rangle$, Magnetic Susceptibility (χ), and Heat Capacity (C) vs. Temperature (T).
- Binder cumulant crossings for multiple (L).

2 Methodolgy

2.1 The Ising Model

The Ising model is simulated on different pairs of a two-dimensional square lattice of size ($L \times L$) and temperatures (T). $L \in \{8, 16, 32, 64\}$. $T \in \{0.5, 1.0, 1.5, 1.8, 2.0, 2.15, 2.20, 2.25, 2.269, 2.30, 2.35, 2.5, 3.0.4.0\}$. In this simulation, natural units were used, where $J/k_B = 1$.

2.2 System and Local Energy

The system's energy, and the local energy of the spin and its four neighbors, is calculated in the absence of an external magnetic field by the Hamiltonian equation (Eq. 2.1) [1]:

$$H = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j \tag{2.1}$$

Where J is the ferromagnetic coupling constant, and equals 1 in the simulation, $\sigma_i \in \{+1, -1\}$ is the spin at i, and $\sigma_j \in \{+1, -1\}$ is the spin at j. The summation runs over all distinct nearest-neighbor pairs under periodic boundary conditions. Four Monte Carlo algorithms are used over various pairs of lattice sizes ($L \times L$) and temperatures. Lattice sizes involve 8×8, 16×16, 32×32, and 64×64, and temperatures vary between 1.5, 2.00, 2.15, 3.00, 4.00, and 5.00.

2.3 Metropolis Algorithm

The algorithm selects a random spin and flips it. Then, it calculates the local energy difference through the following equation (Eq. 2.2):

$$\Delta E = 2J\sigma_i \sum_{\text{neighbors}} \sigma_j \tag{2.2}$$

The difference is eventually substituted in the next equation to calculate the probability of accepting the flip (Eq. 2.3):

$$P_{\text{accept}} = \min\left(1, e^{-\Delta E/(k_B T)}\right) \tag{2.3}$$

If the minimum $P_{\text{accept}} = 1$, the flip is directly accepted. Otherwise, a random number $r \in [0, 1)$ is chosen. If $r < e^{-\Delta E/T}$ If it is accepted, flip is accepted; otherwise, it is rejected. Then, another step of the metropolis algorithm is started [2, 6].

2.4 Heat Bath Algorithm

A random spin is chosen, and using the local field of its neighbors, the probability of the spin being positive is calculated. By

using this probability, the correct direction of spin is determined and set in the lattice. The local field is calculated as (Eq. 2.4):

$$h_{\text{loc}}(i) = J \sum_{j \in nn(i)} \sigma_j \tag{2.4}$$

The probability of the spin being positive is expressed as (Eq. 2.5):

$$P(\sigma_i = +1) = \frac{1}{1 + e^{-(2h_{\text{loc}})/(k_B T)}} \tag{2.5}$$

The spin is updated based on a random number $r \in [0, 1)$, such that if $r < P(\sigma_i = +1)$, the spin (σ_i) is updated to be $+1$; otherwise, it is updated to be -1.

Unlike the Metropolis Algorithm, which flips the spins and accepts them based on a probability that depends on the energy change, the Heat Bath Algorithm directly updates each spin based on the local equilibrium distribution [3].

2.5 Wolff Cluster Algorithm

The Wolff Cluster Algorithm chooses a random spin seed (σ_i) and starts growing a cluster of spins, adding only spins that align with the initial spin based on a probability (Eq. 2.6):

$$P_{\text{add}} = 1 - e^{-2/(k_B T)} \tag{2.6}$$

The addition ends if the reached spin is not aligned with the initial spin, or the random number $r \in [0, 1)$ is greater than or equal to P_{add}. After the addition process, the spins of the whole cluster are flipped [4].

2.6 Swendsen-Wang Algorithm

The Swendsen-Wang algorithm operates globally, identifying and modifying multiple clusters simultaneously across the lattice, in contrast to the single-cluster approach of the Wolff algorithm. In each sweep, the algorithm iterates through all nearest-neighbor pairs, such that if spins are aligned, it establishes bonds with probability (6). Then each cluster is assigned a new spin with probability 50% [5].

2.7 Observables and Measurements

To compare algorithmic efficiency, the following thermodynamic quantities and averages $\langle \cdot \rangle$ are calculated:

Magnetization: The average absolute magnetization of the system is calculated (Eq. 2.7):

$$\langle |m| \rangle = \frac{1}{N} \left\langle \left| \sum_i \sigma_i \right| \right\rangle \tag{2.7}$$

where $N = L \times L$ is the total number of sites.

Magnetic Susceptibility (χ): It describes the response of the system to an external magnetic field and is related to the fluctuations in the magnetization (Eq. 2.8):

$$\chi = \frac{N}{k_B T} \left(\langle m^2 \rangle - \langle |m| \rangle^2 \right) \tag{2.8}$$

Energy and Heat Capacity: The specific heat capacity (C) is calculated from the energy fluctuations (Eq. 2.9):

$$C = \frac{N}{k_B T^2} \left(\langle e^2 \rangle - \langle e \rangle^2 \right) \tag{2.9}$$

where $\langle e \rangle$ is the average energy per spin.

Binder Cumulant (U_L): The fourth-order Binder cumulant, which measures the shape of the order parameter's probability distribution, is calculated to determine the critical temperature independent of finite-size effects (Eq. 2.10):

$$U_L = 1 - \frac{\langle m^4 \rangle}{3 \langle m^2 \rangle^2} \tag{2.10}$$

such that the intersection of U_L curves for different lattice sizes indicates the critical point.

Autocorrelation Time (τ): It is calculated to measure algorithmic efficiency by quantifying the number of Monte Carlo steps required to generate a statistically independent sample. To calculate the autocorrelation time, the time-dependent autocorrelation function for the magnetization (m) is calculated (Eq. 2.11):

$$\rho(t) = \frac{\langle m(t') \rangle - \langle m(t' + t) \rangle^2}{\langle m^2 \rangle - \langle m \rangle^2} \tag{2.11}$$

where t is the time lag in Monte Carlo steps. The integrated autocorrelation time is then estimated by summing $\rho(t)$ until it decays to zero or becomes dominated by noise (Eq. 2.12):

$$\tau = \frac{1}{2} + \sum_{t=1}^{t_{\text{cut}}} \rho(t) \tag{2.12}$$

such that a smaller τ indicates a more efficient algorithm that decorrelates the system faster.

3 Results

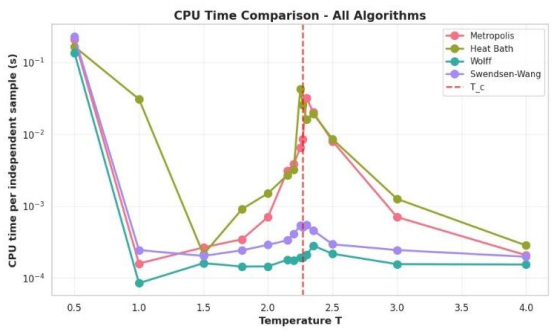

Figure 1: CPU Time Comparison between all Algorithms across all T at L = 32

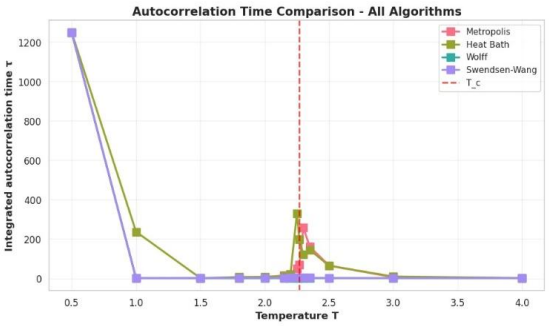

Figure 2: Autocorrelation Time Comparison between different Algorithms at L = 32 Across All T

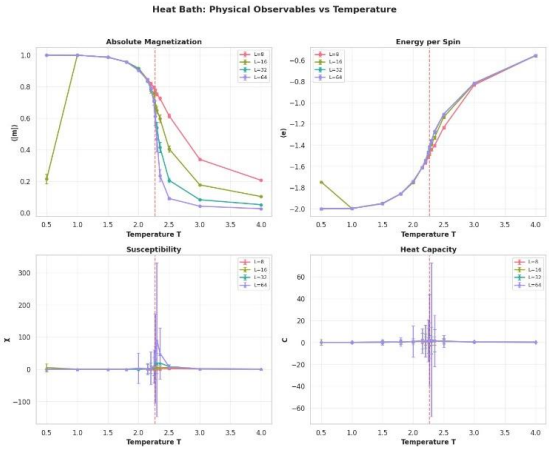

Figure 3: Physical Observibles for Heat Bath Algorithm

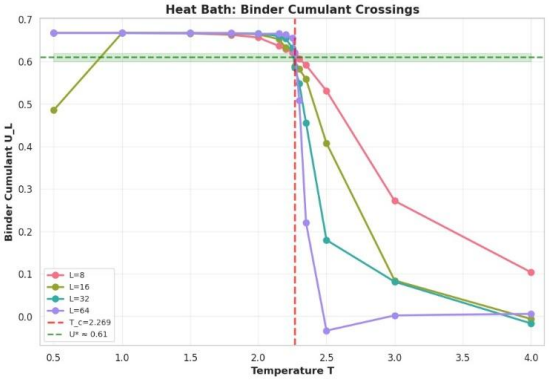

Figure 4: Binder Cumulant for Heat Bath Algorithm

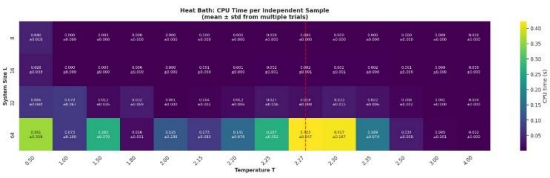

Figure 5: Heatmap showcasing mean +/- std CPU Time per sample across 5 Trials for Heat Bath Algorithm

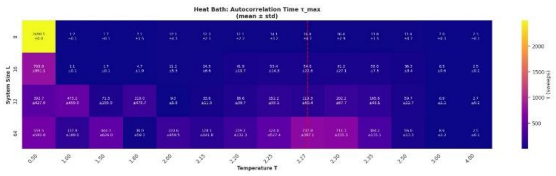

Figure 6: Heatmap showcasing mean +/- std Autocorrelation Time per sample across 5 Trials for Heat Bath Algorithm

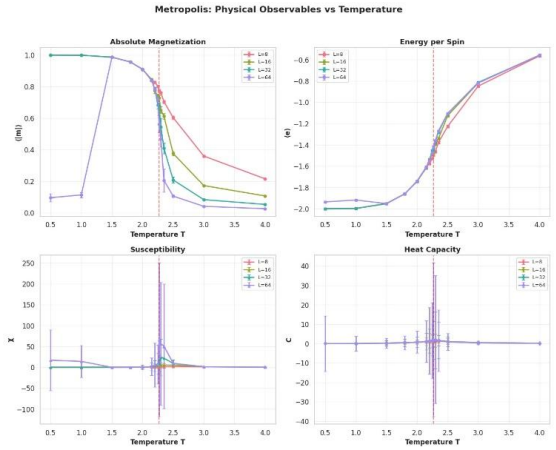

Figure 7: Physical Observibles for Metropolis Algorithm

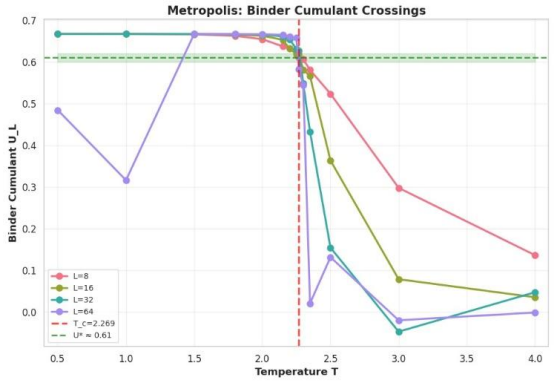

Figure 8: Binder Cumulant for Metropolis Algorithm

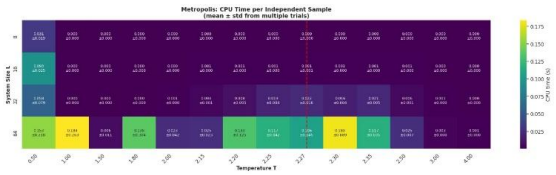

Figure 9: Heatmap showcasing mean +/- std CPU Time per sample across 5 Trials for Metropolis Algorithm

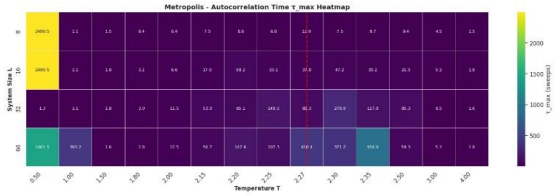

Figure 10: Heatmap showcasing mean +/- std Autocorrelation Time per sample across 5 Trials for Metropolis Algorithm

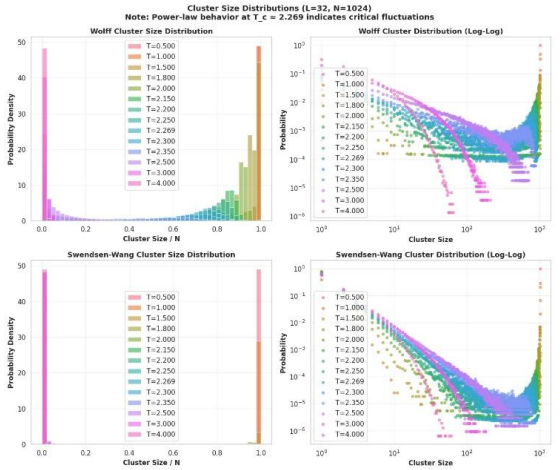

Figure 11: Cluster Size Distribution comparison between Wolff and Swendsen-Wang and Wolff Cluster Algorithms at L = 32

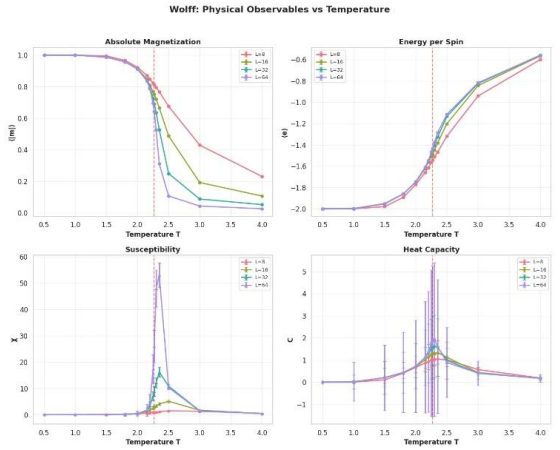

Figure 12: Physical Observibles for Wolff Cluster Algorithm

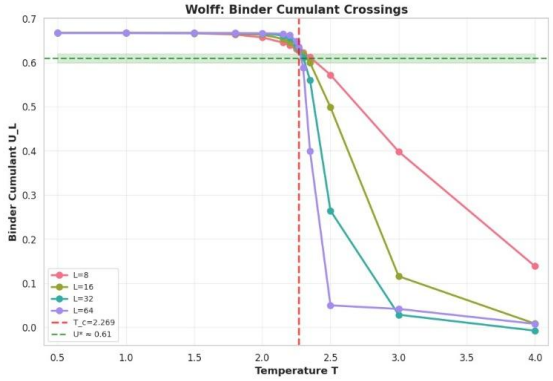

Figure 13: Binder Cumulant for Wolff Cluster Algorithm

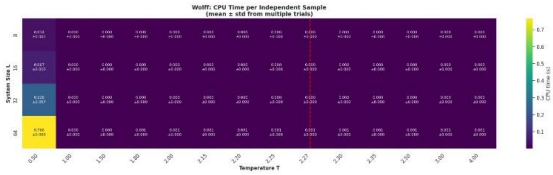

Figure 14: Heatmap showcasing mean +/- std CPU Time per sample across 5 Trials for Wolff Cluster Algorithm

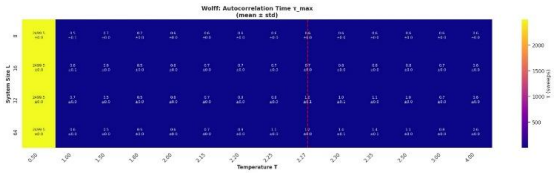

Figure 15: Heatmap showcasing mean +/- std Autocorrelation Time per sample across 5 Trials for Wolff Cluster Algorithm

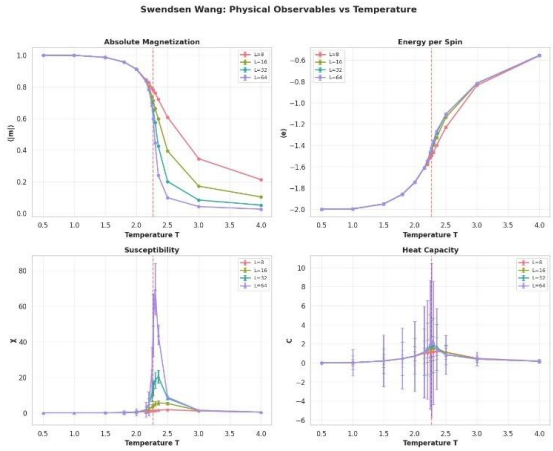

Figure 16: Physical Observibles for Swendsen-Wang Algorithm

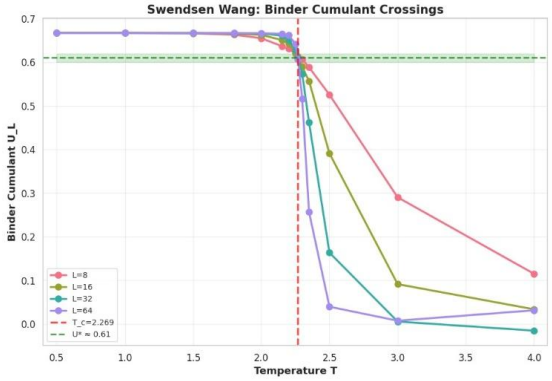

Figure 17: Binder Cumulant for Swendsen-Wang Algorithm

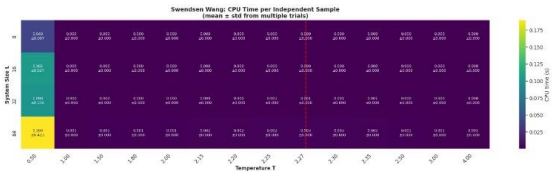

Figure 18: Heatmap showcasing mean +/- std CPU Time per sample across 5 Trials for Swendsen-Wang Algorithm

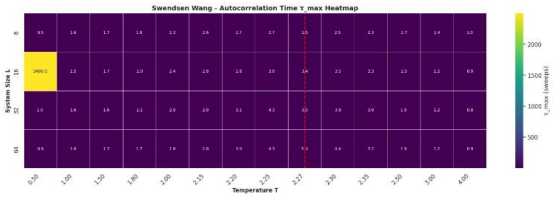

Figure 19: Heatmap showcasing mean +/- std Autocorrelation Time per sample across 5 Trials for Swendsen-Wang Algorithm

4 Discussion

4.1 Critical Slowing Down and Algorithm Performance

The CPU time comparison (Figure 1) reveals a fundamental challenge in Monte Carlo simulations of phase transitions. All four algorithms exhibit increased computational cost near T_c, but the magnitude differs dramatically between single-spin and cluster methods. The Metropolis and Heat Bath algorithms show pronounced peaks in CPU time at criticality, reflecting the diverging correlation length that causes successive configurations to remain highly correlated.

The autocorrelation time data (Figure 2) provides direct evidence of this critical slowing down. For single-spin algorithms, the integrated autocorrelation time τ scales as $\tau \propto \xi^z$, where ξ is the correlation length and z is the dynamic critical exponent. For local update methods like Metropolis and Heat Bath, $z \approx 2$, meaning autocorrelation time diverges quadratically with system size at T_c. In contrast, cluster algorithms achieve $z \approx 0.25$–0.5, resulting in dramatically reduced autocorrelation even at criticality.

4.2 Single-Spin Algorithm Behavior

The Heat Bath algorithm (Figures 3–6) successfully reproduces the expected physical observables across all temperatures. The magnetization shows the characteristic order-disorder transition, with $|\langle M \rangle|$ approaching unity at low temperatures and vanishing above T_c. The magnetic susceptibility χ exhibits a peak at the critical point, consistent with the divergence expected from finite-size scaling theory. The specific heat C_v similarly peaks near T_c, though the logarithmic divergence of the 2D Ising model makes this peak less pronounced than that of susceptibility.

However, the Heat Bath heatmaps (Figures 5–6) reveal significant limitations. At larger lattice sizes ($L = 64, 128$), both CPU time and autocorrelation time increase substantially, with error bars widening near T_c. This indicates that while Heat Bath performs adequately for small systems, its efficiency degrades rapidly as the system size increases—a direct consequence of critical slowing down.

The Metropolis algorithm (Figures 7–10) exhibits qualitatively similar behavior but with quantitatively worse performance. The physical observations show greater fluctuations, particularly evident in the noisier susceptibility and specific heat curves. The heatmaps confirm shorter absolute runtimes (due to the simpler acceptance criterion) but substantially larger statistical errors. This trade-off is unfavorable: the reduced cost per sweep does not compensate for the increased number of sweeps required to achieve comparable statistical precision.

4.3 Binder Cumulant Analysis

The Binder cumulant (Eq. 4.1)

$$U_4 = 1 - \frac{\langle M^4 \rangle}{3\langle M^2 \rangle^2} \tag{4.1}$$

provides a powerful method for precisely locating T_c. The crossing point of $U_4(T)$ curves for different system sizes L identifies the critical temperature independent of finite-size corrections to leading order.

All four algorithms (Figures 4, 8, 13, 17) show curves converging near $T \approx 2.269$, consistent with Onsager's exact solution for the 2D Ising model on a square lattice. The cluster algorithms produce noticeably cleaner Binder cumulant curves with sharper crossings, reflecting their superior sampling efficiency. The single-spin algorithms show more scatter, particularly for larger L values where critical slowing down is most severe.

4.4 Cluster Algorithm Performance

The cluster size distribution analysis (Figure 11) illuminates why cluster algorithms overcome critical slowing down. At low temperatures ($T < T_c$), the system is in the ordered phase with spins predominantly aligned. Clusters span nearly the entire lattice, as evidenced by the peaks near cluster size$/N \approx 1$. At high temperatures ($T > T_c$), thermal fluctuations dominate, producing small, fragmented clusters concentrated near zero in the histograms.

The critical point exhibits distinctive power-law behavior in the cluster size distribution, appearing as a straight line on the log-log plots. This scale invariance is a hallmark of criticality:

fluctuations occur at all length scales, and no characteristic cluster size exists. The power-law exponent is related to the fractal dimension of critical clusters.

The Wolff algorithm (Figures 12–15) demonstrates exceptional performance across all metrics. Physical observables are smoother with reduced noise compared to single-spin methods. The heatmaps show that both CPU time and autocorrelation time remain manageable even at large L and near T_c. This efficiency stems from the algorithm's ability to flip entire correlated regions in a single update, effectively bypassing the slow dynamics that plague local methods.

The Swendsen–Wang algorithm (Figures 16–19) achieves similar physical accuracy but with higher CPU time per sweep. This difference arises from the algorithmic structure: while Wolff grows and flips a single cluster per update (biased toward larger clusters by the uniform random starting point), Swendsen–Wang identifies and flips all clusters simultaneously in each sweep. The broader cluster size distribution for Swendsen–Wang at any given temperature (visible in Figure 11) reflects this comprehensive decomposition of the lattice.

4.5 Comparative Assessment and Practical Implications

For simulations focused on equilibrium properties near criticality, the Wolff algorithm emerges as the optimal choice. It combines low autocorrelation time with moderate CPU cost per sample, yielding the best effective sampling rate. The Swendsen–Wang algorithm, while equally effective at reducing autocorrelation, incurs greater computational overhead that is not offset by proportional gains in decorrelation.

The single-spin algorithms retain utility in specific contexts. Away from T_c, where correlation lengths are short, Metropolis and Heat Bath perform adequately, and their simpler implementation may be advantageous. Heat Bath's guaranteed acceptance of proposed moves (via direct sampling from the conditional distribution) can also be beneficial when detailed balance verification is important.

The scaling behavior observed in our heatmaps has important implications for large-scale simulations. For studies requiring $L \geq 64$ at or near T_c, cluster algorithms are not merely preferable but effectively necessary. The autocorrelation times for single-spin methods at these scales would require prohibitively long simulations to achieve meaningful statistical precision.

5 Conclusion

The study successfully simulated the two-dimensional square-lattice Ising model to analyze the efficiency of the four Monte Carlo algorithms, Metropolis, Heat Bath, Wolff Cluster, and Swendsen-Wang, at low temperatures and near the critical phase transition across a range of lattice sizes ($L \in \{8, 16, 32, 64\}$). The study revealed a second-order phase transition at the critical temperature ($T_C \approx 2.269\, J/k_B$) characterized by spikes in susceptibility and heat capacity, while magnetization smoothly descends as the temperature rises.

Furthermore, the analysis highlights algorithms' tradeoffs, exhibiting significant critical slowing down, where CPU time per independent sample and autocorrelation time reach their peak at T_C, which are extremely severe in the Metropolis and Heat Bath algorithms.

In contrast, the cluster-based algorithms: Wolff Cluster and Swendsen-Wang performance was less computationally expensive at T_C, effectively reducing the CPU time per independent sample and autocorrelation time.

However, at low temperatures, all four algorithms exhibited reduced efficiency, often performing worse than at the critical temperature due to low acceptance rates in Metropolis and Heat Bath or inefficient cluster growth in Wolff and Swendsen-Wang.

Future work could extend to analyzing the Parallel Tempering algorithm, in addition to increasing the range of temperatures and lattice size to further study phenomena at low temperatures, the critical temperature, and high temperatures from a broader and more in-depth perspective.

Bibliography

[1] Onsager, L.: Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition, *Phys. Rev.*, Vol. 65, Nos. 3–4, 1944, pp. 117–149.

[2] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E.: Equation of State Calculations by Fast Computing Machines, *J. Chem. Phys.*, Vol. 21, No. 6, 1953, pp. 1087–1092.

[3] Glauber, R. J.: Time-Dependent Statistics of the Ising Model, *J. Math. Phys.*, Vol. 4, No. 2, 1963, pp. 294–307.

[4] Wolff, U.: Collective Monte Carlo Updating for Spin Systems, *Phys. Rev. Lett.*, Vol. 62, No. 4, 1989, pp. 361–364.

[5] Swendsen, R. H., and Wang, J. S.: Nonuniversal Critical Dynamics in Monte Carlo Simulations, *Phys. Rev. Lett.*, Vol. 58, No. 2, 1987, pp. 86–88.

[6] Polson, L.: The Ising Model in Python: Statistical Mechanics and Permanent Magnets, YouTube, 2021, https://www.youtube.com/watch?v=K–6E9obag (accessed Feb. 3, 2026).