Studio Numerico della Transizione di Fase del Modello di Ising 2D mediante Algoritmo di Wolff

Federico Spinello

17 gennaio 2026

Sommario

In questo lavoro vengono studiate numericamente la transizione di fase ferromagnetica del modello di Ising bidimensionale su reticolo quadrato mediante simulazioni Monte Carlo. Viene utilizzato l'algoritmo di Wolff per superare il fenomeno del rallentamento critico che affligge gli algoritmi locali come Metropolis. Attraverso l'analisi di Finite Size Scaling per reticoli di dimensioni $L = 40, 60, 80, 100, 120, 140, 160, 180, 200$ e temperature nel range $T \in [2.1, 2.4]$ (100 punti), determiniamo la temperatura critica tramite il crossing del Binder cumulant e l'analisi FSS dei picchi della suscettività e del calore specifico, ottenendo, rispettivamente: $T_c^B = 2.2690 \pm 0.0001$; $T_c^\chi = 2.267 \pm 0.001$ ($\chi^2_{\text{red}} = 0.037$) e $T_c^C = 2.265 \pm 0.004$ ($\chi^2_{\text{red}} = 0.16$); l'esponente critico tramite scaling della suscettività $\gamma/\nu = 1.750 \pm 0.005$ ($\chi^2_{\text{red}} = 0.86$), in buon accordo con i valori esatti noti dalla soluzione di Onsager ($T_c = 2.2692$ e $\gamma/\nu = 1.75$). Gli errori sui picchi sono stati stimati mediante metodo bootstrap non parametrico. Il data collapse conferma ulteriormente la validità della teoria FSS. I risultati dimostrano l'efficacia dell'algoritmo di Wolff e l'accuratezza dei metodi Monte Carlo per lo studio delle transizioni di fase.

1 Introduzione

Il modello di Ising rappresenta uno dei paradigmi fondamentali della meccanica statistica. Nonostante la sua semplicità—variabili di spin discrete su reticolo con interazioni tra primi vicini—cattura la fisica delle transizioni di fase di secondo ordine.

Nel 1944, Lars Onsager ottenne la soluzione esatta del modello di Ising bidimensionale, determinando la temperatura critica

$$T_c = \frac{2J}{k_B \ln\bigl(1+\sqrt{2}\bigr)} \simeq 2.2692 \tag{1}$$

Un problema notevole è il *rallentamento critico*: vicino a T_c, algoritmi locali come Metropolis diventano inefficienti. L'algoritmo di Wolff (1989) risolve questo problema usando aggiornamenti dei cluster.

1.1 Obiettivi

Questo lavoro si propone di:

1. Determinare T_c tramite picchi di suscettività, calore specifico, e il crossing del Binder cumulant
2. Stimare l'esponente critico γ/ν
3. Validare l'effetto della taglia finita (Finite Size Scaling, o FSS per brevità) attraverso il collasso dei dati

2 Il Modello di Ising 2D

2.1 Definizione

Il modello è definito su un reticolo quadrato Λ di dimensione $L \times L$ con $N = L^2$ siti. A ciascun sito $i \in \Lambda$ è associata una variabile di spin:

$$s_i \in \{-1, +1\} \tag{2}$$

L'Hamiltoniana ferromagnetica è:

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} s_i s_j \tag{3}$$

dove $J > 0$ (poniamo $J = 1$) e $\langle i, j \rangle$ indica coppie di primi vicini. Usiamo condizioni periodiche al contorno (PBC).

2.2 Transizione di Fase

Il sistema presenta una transizione di fase di secondo ordine alla temperatura critica T_c:

- **Fase ferromagnetica** ($T < T_c$): Magnetizzazione spontanea $m \neq 0$, ordine a lungo range.
- **Fase paramagnetica** ($T > T_c$): Magnetizzazione $m = 0$, spin scorrelati.

A T_c emerge un comportamento critico caratterizzato dalla divergenza della *lunghezza di correlazione* $\xi(T)$. La lunghezza di correlazione è la distanza caratteristica oltre la quale gli spin perdono memoria dell'orientazione

reciproca. Più precisamente, la funzione di correlazione spin-spin decade esponenzialmente:

$$\langle s_i s_j \rangle \sim e^{-|r_i - r_j|/\xi(T)} \tag{4}$$

dove $|r_i - r_j|$ è la distanza tra i siti i e j.

Vicino alla temperatura critica, ξ diverge secondo la legge di potenza:

$$\xi(T) \sim |T - T_c|^{-\nu} \tag{5}$$

con $\nu = 1$ per Ising 2D. Nella fase ferromagnetica ($T < T_c$), ξ rappresenta la dimensione tipica dei domini ordinati; nella fase paramagnetica ($T > T_c$), misura la distanza su cui gli spin rimangono correlati prima di disordinarsi. Esattamente a T_c, $\xi = \infty$: il sistema presenta correlazioni a lungo raggio e strutture a tutte le scale (invarianza di scala).

2.3 Esponenti Critici

Il comportamento vicino a T_c è governato da:

$$m(T) \sim (T_c - T)^{\beta}, \qquad \beta = 1/8 \tag{6}$$

$$\chi(T) \sim |T - T_c|^{-\gamma}, \qquad \gamma = 7/4 = 1.75 \tag{7}$$

$$\xi(T) \sim |T - T_c|^{-\nu}, \qquad \nu = 1 \tag{8}$$

$$C(T) \sim -\ln|T - T_c|, \qquad \alpha = 0 \tag{9}$$

Questi valori sono esatti (Onsager) e universali: dipendono solo da dimensionalità ($d = 2$), simmetria ($\mathbb{Z}_2$) e range delle interazioni.

2.4 Finite Size Scaling

Su un reticolo finito, ξ non può superare L. La teoria di Finite Size Scaling (FSS) descrive come le osservabili termodinamiche si comportano in sistemi di dimensione finita vicino alla transizione di fase.

Consideriamo un'osservabile generica $O(T, L)$ che dipende sia dalla temperatura T che dalla dimensione del sistema L (ad esempio, magnetizzazione, suscettività, energia, etc.). La teoria FSS prevede che questa osservabile possa essere fattorizzata come:

$$O(T, L) = L^{\lambda/\nu} \tilde{O}\left(L^{1/\nu}(T - T_c)\right) \tag{10}$$

dove:

- $O(T, L)$ è il valore misurato dell'osservabile su un reticolo finito $L \times L$
- $L^{\lambda/\nu}$ è un fattore di scala che dipende dall'esponente critico λ dell'osservabile e dall'esponente ν della lunghezza di correlazione
- $\tilde{O}(x)$ è una *funzione di scaling universale*, indipendente da L e T separatamente, che dipende solo dalla combinazione adimensionale $x = L^{1/\nu}(T - T_c)$

La potenza di questa relazione è che tutte le curve $O(T, L)$ per diversi valori di L, se riscalate opportunamente, collassano su un'unica curva universale $\tilde{O}(x)$. Questo è il fenomeno del *data collapse*.

Per la suscettività ($\lambda = \gamma$):

$$\chi(T, L) = L^{\gamma/\nu} \tilde{\chi}\left(L^{1/\nu}(T - T_c)\right) \tag{11}$$

Conseguenze:

- Picco a $T_{\max}(L) \approx T_c + \text{const} \cdot L^{-1/\nu}$
- Altezza: $\chi_{\max}(L) \sim L^{\gamma/\nu}$
- Data collapse: rappresentando $\chi/L^{\gamma/\nu}$ su $L^{1/\nu}(T - T_c)$, tutte le curve collassano nella stessa

2.5 Binder Cumulant

Il Binder cumulant è definito come:

$$U_L = \frac{\langle m^4 \rangle}{\langle m^2 \rangle^2} \tag{12}$$

Proprietà:

- $T \ll T_c$: $U_L \to 1$ (fase ordinata)
- $T \gg T_c$: $U_L \to 3$ (fase disordinata, fluttuazioni gaussiane)
- $T = T_c$: $U_L = U^* \approx 1.17$ (universale per Ising 2D)

Le curve $U_L(T)$ per diversi L si intersecano a T_c (*crossing point*).

3 Metodi Monte Carlo

3.1 Il Problema Computazionale

In meccanica statistica, il valore medio di un'osservabile nell'ensemble canonico richiede:

$$\langle O \rangle = \frac{1}{Z} \sum_{\{s\}} O(\{s\}) e^{-\beta \mathcal{H}(\{s\})} \tag{13}$$

dove $Z = \sum_{\{s\}} e^{-\beta \mathcal{H}}$ è la funzione di partizione.

Per $N = L^2$ spin, esistono 2^N configurazioni possibili. Esempi numerici:

- $L = 10$: $2^{100} \approx 10^{30}$ configurazioni
- $L = 20$: $2^{400} \approx 10^{120}$ configurazioni (più atomi nell'universo!)
- $L = 100$: $2^{10000} \approx 10^{3010}$ configurazioni

Il calcolo esatto è **impossibile** per sistemi realistici.

3.2 Importance Sampling

La soluzione Monte Carlo: invece di sommare su tutte le configurazioni, si campionano solo quelle importanti secondo la distribuzione di Boltzmann:

$$P(\{s\}) = \frac{1}{Z} e^{-\beta \mathcal{H}(\{s\})} \tag{14}$$

Con M configurazioni campionate correttamente:

$$\langle O \rangle \approx \frac{1}{M} \sum_{i=1}^{M} O(\{s^{(i)}\}) \tag{15}$$

Con $M = 4 \times 10^5$ misure si ottiene precisione statistica eccellente, anche se $M \ll 2^N$.

3.3 Algoritmo di Metropolis

L'algoritmo di Metropolis (1953) genera configurazioni secondo $P(\{s\})$ usando una catena di Markov:

1. Scegli un sito casuale (i, j)
2. Proponi l'inversione: $s_{ij} \to -s_{ij}$
3. Calcola $\Delta E = E_{\text{nuova}} - E_{\text{vecchia}}$
4. Accetta con probabilità:

$$P_{\text{acc}} = \min\{1, e^{-\beta \Delta E}\} \tag{16}$$

Questo garantisce il bilancio dettagliato e convergenza a $P(\{s\})$.

3.4 Rallentamento critico

Vicino a T_c, Metropolis diventa estremamente inefficiente. Questo perché:

- La lunghezza di correlazione diverge: $\xi \sim |T - T_c|^{-\nu}$
- Si formano cluster di spin correlati di dimensione $\sim \xi$
- Metropolis inverte un singolo spin per volta
- Per decorrelazione servono $\tau \sim \xi^z$ passi, con $z \approx 2$ (dinamica locale)

Risultato: a T_c su reticolo $L \times L$:

$$\tau_{\text{Metropolis}} \sim L^2 \tag{17}$$

Per $L = 100$: servono ~ 10000 volte più passi che per $L = 10$.

3.5 Algoritmo di Wolff

Wolff (1989) risolve il rallentamento critico aggiornando direttamente un cluster: invece di invertire singoli spin, inverte interi cluster di spin correlati.

3.5.1 Costruzione del Cluster

1. Scegli seme s_0 casuale
2. Inizializza cluster: $C = \{s_0\}$, occupati $= \{s_0\}$
3. Calcola probabilità: $P_{\text{add}} = 1 - e^{-2\beta}$
4. Per ogni sito $s \in C$:
   - Esamina i 4 primi vicini
   - Se il vicino ha *stesso spin* e *non è occupato*: aggiungi a C con prob. P_{add}
5. Ripeti finché nessun nuovo sito viene aggiunto
6. **Flippa tutto il cluster** C simultaneamente

3.5.2 Bilancio dettagliato

Wolff dimostrò che questa procedura rispetta il bilancio dettagliato con $P_{\text{add}} = 1 - e^{-2\beta}$. La probabilità di costruire un cluster C e il suo complemento $\bar{C}$ sono identiche.

3.5.3 Efficienza

Il vantaggio cruciale: a T_c, dove esistono cluster naturali di dimensione $\sim L$, Wolff li flippa in un singolo step, questo rende questo algoritmo molto più veloce di Metropolis.

Tempo di autocorrelazione:

$$\tau_{\text{Metropolis}} \sim L^z, \qquad z \approx 2 \tag{18}$$

$$\tau_{\text{Wolff}} \sim L^{z_W}, \qquad z_W \approx 0.25 \tag{19}$$

Speed-up per $L = 100$:

$$\frac{\tau_{\text{Metropolis}}}{\tau_{\text{Wolff}}} \approx \frac{100^2}{100^{0.25}} \approx \frac{10000}{3.16} \approx 3160 \tag{20}$$

Wolff è **oltre 3000 volte più veloce** vicino a T_c!

3.6 Implementazione Non-Ricorsiva

Per evitare stack overflow su reticoli grandi, implementiamo Wolff in modo iterativo usando una coda di siti da esplorare (`pointtoocc`). Questo permette di costruire cluster fino a $L \sim 1000$ senza problemi di memoria.

4 Dettagli Simulazione

4.1 Parametri

Tabella 1: Parametri delle simulazioni

Parametro	**Valore**
Dimensioni L	40, 60, 80, 100, 120, 140, 160, 180, 200
Temperature T	2.1–2.4 (100 punti)
Termalizzazione	10000 aggiornamenti dei cluster
Misure	400000 aggiornamenti dei cluster

4.2 Osservabili

Le osservabili che vengono analizzate sono le seguenti:

Magnetizzazione (valore assoluto medio) La magnetizzazione istantanea è definita come:

$$m = \frac{1}{volume} \sum_i s_i \tag{21}$$

Per studiare la transizione di fase, misuriamo il valore assoluto medio $\langle |m| \rangle$ ovvero:

1. ad ogni step misuriamo m
2. prendiamo il valore assoluto
3. cluster update
4. ripeti fino a 4×10^5 passi
5. media tutti i valori ottenuti

Energia per spin

$$E = -\frac{1}{N} \sum_{\langle i,j \rangle} s_i s_j \tag{22}$$

Suscettività

$$\chi = \beta N \left(\langle m^2 \rangle - \langle |m| \rangle^2 \right) \tag{23}$$

Binder cumulant Il Binder cumulant (o quarto cumulante) è una quantità adimensionale particolarmente utile per determinare la temperatura critica in sistemi finiti. È definito come:

$$U_L = \frac{\langle m^4 \rangle}{\langle m^2 \rangle^2} \tag{24}$$

Questa quantità ha proprietà notevoli che la rendono ideale per l'analisi FSS:

Comportamento limite:

- **Fase ordinata** ($T \ll T_c$): La magnetizzazione è quasi costante, $m \approx m_0$. Quindi $\langle m^4 \rangle \approx \langle m^2 \rangle^2$ e $U_L \to 1$.
- **Fase disordinata** ($T \gg T_c$): Le fluttuazioni sono gaussiane. Per una distribuzione gaussiana, $\langle m^4 \rangle = 3\langle m^2 \rangle^2$, quindi $U_L \to 3$.

Il Binder cumulant viene calcolato accumulando durante la simulazione i momenti $\langle m^2 \rangle$ e $\langle m^4 \rangle$ ad ogni aggiornamento dei cluster e infine, calcolando il rapporto.

Calore specifico Il calore specifico è definito come la derivata dell'energia rispetto alla temperatura:

$$C = \frac{\partial \langle E \rangle}{\partial T} \tag{25}$$

e viene calcolato tramite il teorema di fluttuazione-dissipazione:

$$C = \beta^2 N \left(\langle E^2 \rangle - \langle E \rangle^2 \right) \tag{26}$$

4.3 Termalizzazione e Analisi degli Errori

Scelta del periodo di termalizzazione Prima di raccogliere misure statisticamente significative, il sistema deve raggiungere l'equilibrio termico partendo da una configurazione iniziale (in questo caso, completamente ordinata con tutti gli spin +1).

In questo caso sono stati eseguiti 10000 cluster update, il motivo è puramente sperimentale, ovvero sono stati tracciati i grafici della magnetizzazione e energia per ogni step temporale, nella configurazione che richiedeva più tempo per essere termalizzata, ovvero il caso in cui $L = 200$ e $T = 2.4$.

1. È stato osservato che l'energia raggiunge un plateau stazionario dopo $\sim$ 1000 aggiornamenti dei cluster, con fluttuazioni statistiche attorno al valore medio.
2. La magnetizzazione, invece, si stabilizzava molto più velocemente.
3. Per sicurezza, è stato scelto un periodo di termalizzazione di 10000 step, ovvero un ordine di grandezza in più, garantendo che il sistema sia completamente termalizzato.

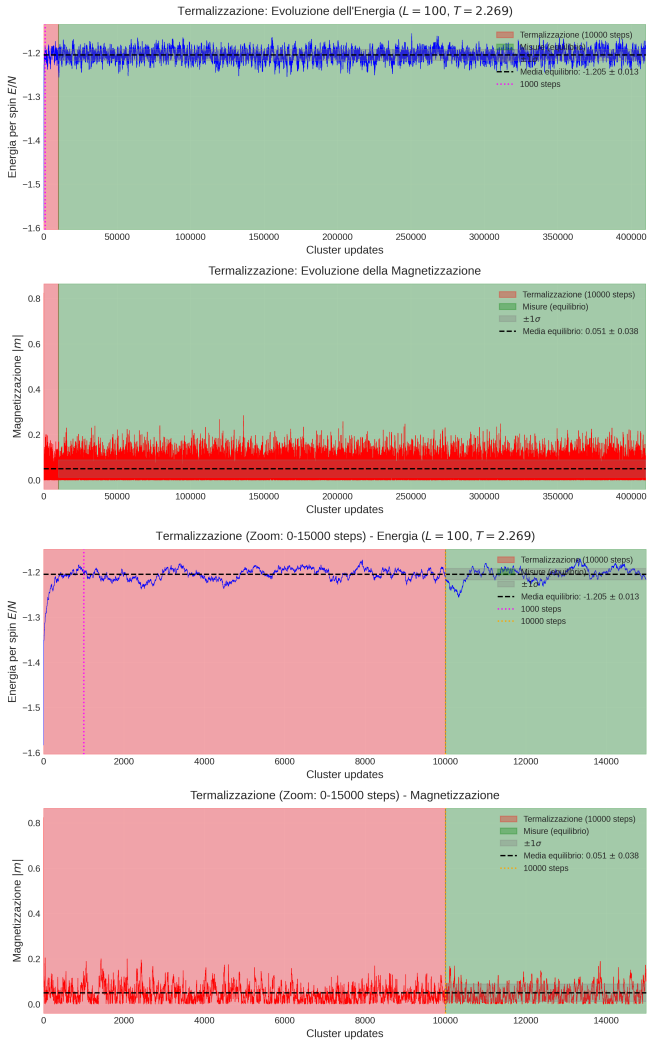

Figura 1: Esempio di termalizzazione per $L = 200$ a $T = 2.4$. Le regioni evidenziate mostrano la fase di termalizzazione (primi 10000 steps, in rosso) e la fase di misura in equilibrio (steps successivi, in verde). Dopo ~ 1000 (linea verticale viola) step l'energia ha raggiunto un regime stazionario, la magnetizzazione, invece, lo raggiunge molto prima. In basso uno zoom delle prime 15000 misurazioni

Autocorrelazione e dimensione dei blocchi L'algoritmo di Wolff riduce drasticamente il tempo di autocorrelazione τ rispetto a Metropolis. Per Ising 2D, è noto dalla letteratura che $\tau_{\text{Wolff}} \sim L^{z_W}$ con $z_W \approx 0.25$. Per $L = 100$, questo corrisponde a $\tau \sim 100^{0.25} \approx 3$ aggiornamenti dei cluster.

Con 400000 misure raccolte dopo la termalizzazione, il numero di configurazioni statisticamente indipendenti è:

$$N_{\text{indep}} \approx \frac{400000}{\tau} \sim \frac{400000}{3-10} \sim 10^5 \tag{27}$$

Questo numero è sufficientemente grande da garantire errori statistici piccoli sulle medie. Il numero elevato di misure e il basso tempo di autocorrelazione dell'algoritmo di Wolff rendono gli errori statistici trascurabili rispetto alle incertezze sistematiche dovute agli effetti di finite size.

Stima degli errori Gli errori riportati in questo lavoro hanno diverse origini:

- **Errori sui singoli punti**: Mostrati esplicitamente solo in alcuni grafici per chiarezza visiva, ma tipicamente $< 1\%$ del valore misurato grazie all'elevato numero di misure.
- **Parametri da fit** (T_c, γ/ν): Errori stimati da `scipy.optimize.curve_fit`, che propagano l'incertezza statistica sui dati ai parametri fittati.
- **Binder crossing**: L'errore è stimato come la semilarghezza della regione di temperatura dove la dispersione $\sigma(U_L)$ rimane entro il 5% del minimo. Riflette la dispersione residua tra le curve di diversi L.

4.4 Implementazione

4.5 Architettura del Software

Particolare importanza è stata data alla progettazione dell'architettura del software. Questa è modulare, separando nettamente le fasi di simulazione (C), analisi (Python) e presentazione (LaTeX). Questa separazione segue il principio della *separazione dei compiti* e permette di eseguire ri-esecuzioni parziali e una maggiore manutenibilità.

Design Pattern L'intero flusso di lavoro è organizzato come una successione di 4 stadi:

1. **Parallelizzazione**: Codice python che gestisce l'organizzazione delle simulazioni
2. **Compilazione**: Codice C compilato con diverse ottimizzazioni
3. **Simulazione**: Esecuzione Monte Carlo su griglia
4. **Analisi**: Post-processing Python con calcolo esponenti e grafici

Ogni stadio comunica tramite *file di dati*, permettendo riesecuzioni parziali e debugging indipendente.

Parallelizzazione Le simulazioni Monte Carlo sono computazionalmente intensive: una configurazione, ad esempio utilizzando: $L = 80, 100$ con 100 temperature e 400000 misurazioni ciascuna richiede diverse ore di calcolo sequenziale. Poiché ogni coppia (L, T) rappresenta una simulazione completamente indipendente, il problema è facilmente parallelizzabile.

Decomposizione del problema: Il gruppo di simulazioni $(L_1, \ldots, L_n) \times (T_1, \ldots, T_m)$ viene decomposto in $n \times m$ lavori indipendenti. Ogni lavoro (L_i, T_j) è una singola simulazione isolata che produce il file `data/L{L}_T{T}.dat`.

Architettura thread-safe: Lo script `parallel_run.py` implementa un bacino di lavoratori tramite `multiprocessing.Pool`, creando un processo per ogni core CPU disponibile (in questo caso 16). Ogni lavoratore opera in completo isolamento:

1. Crea una cartella temporanea unica
2. Scrive un file `params.txt` locale con parametri della singola simulazione, ad esempio: `L = 40`, `T = 2.100`, `OUTPUT_FILE = data/L40_T2.1000.dat`
3. Esegue il binario `bin/ising_simulation` dalla cartella temporanea
4. Il binario legge i parametri e invoca direttamente `run_ising_simulation()` una volta
5. Scrive i dati nel file specificato
6. Rimuove la cartella temporanea al termine

Questa architettura elimina completamente i conflitti: ogni lavoratore ha il proprio file di configurazione isolato, mentre i file in uscita hanno nomi univoci per costruzione (`L{L}_T{T:.4f}.dat`).

Prestazioni: Con 16 core CPU e `MEASUREMENTS=400000`, una configurazione di 700 simulazioni (L che va da 80 a 200, con passo 20 $\times$ 100 temperature) ha richiesto $\sim$ 176 minuti ($\sim$ 3h) invece di circa 2 giorni sequenziali, ottenendo una velocizzazione di $\sim$16$\times$. Il bacino di lavoratori distribuisce dinamicamente i lavori sui core disponibili, massimizzando l'utilizzo delle risorse.

Modularizzazione del Codice C Il simulatore C è diviso in moduli funzionali:

- `main.c`: Involucro minimale per singola simulazione
  - Legge `params.txt` con parametri singolo lavoro (L, T, passi)
  - Invoca `run_ising_simulation()` una volta
- `ising_wolff.c/h`: Nucleo della simulazione fisica
  - Implementazione algoritmo di Wolff (aggiornamenti dei cluster)
  - Ciclo Monte Carlo (termalizzazione + misure)
  - Costruzione cluster non ricorsiva
  - Calcolo osservabili (m, E, χ, C, Binder)
- `geometry.c/h`: Costruzione griglia
  - Condizioni contorno periodiche
  - Calcolo indici vicini (N, S, E, O)
  - Mappatura 2D $\to$ 1D: $(i, j) \to i \cdot L + j$
- `random.c/h`: Interfaccia generatore numeri casuali
  - Involucro uniforme per PCG32
  - Inizializzazione con seme temporale
- `pcg32min.c/h`: Generatore PCG32
  - Implementazione algoritmo PCG (Permuted Congruential Generator)

Questa architettura separa l'involucro minimale (main.c) dalla logica fisica (ising_wolff.c).

Automazione con Makefile Il `Makefile` implementa target dichiarativi per l'intero processo:

- `make all`: Compilazione con dipendenze automatiche
- `make run`: Esegue la simulazione in parallelo su tutti i core CPU
- `make analyze`: Esegue `analyze.py` $\to$ genera i grafici e i risultati e li salva in /plots e risultati.txt
- `make paper`: Compilazione LaTeX
- `make full`: Intero flusso di lavoro, dalla generazione dei dati, all'analisi

Configurazione Parametrica Per una maggiore manutenbilità e chiarezza i parametri iniziali si trovano in `params.txt`:

- `L_VALUES`: Lista dimensioni reticolo
- `T_MIN`, `T_MAX`, `N_TEMPS`: Griglia temperature
- `THERMALIZATION`, `MEASUREMENTS`: Numero aggiornamenti MC
- `DATA_DIR`: Cartella di dati

L'eseguibile `bin/ising_simulation` legge `params.txt` all'avvio, costruisce la griglia ($L \times T$) e lancia tutte le simulazioni.

Post-Processing Python `analyze.py` implementa l'intera analisi dei dati:

- Caricamento dati dai file .dat
- Calcolo Binder crossing
- Fit della legge di potenza $\chi_{\max} \sim L^{\gamma/\nu}$ con `scipy.curve_fit`
- Collasso dei dati dovuto al FSS
- Generazione dei grafici
- Salvataggio dei risultati in `risultati.txt`

5 Risultati

5.1 Magnetizzazione

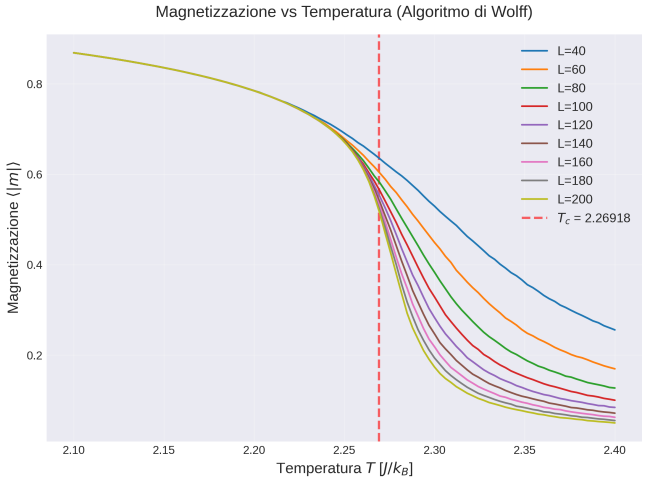

Figura 2: Magnetizzazione vs temperatura. La transizione diventa più ripida con L.

La Figura 2 mostra:

- $T < T_c$: magnetizzazione spontanea (fase ordinata)
- $T \approx T_c$: transizione rapida
- $T > T_c$: $m \to 0$ (fase disordinata)
- La transizione diventa più ripida all'aumentare di L (effetto della taglia finita)

5.2 Suscettività

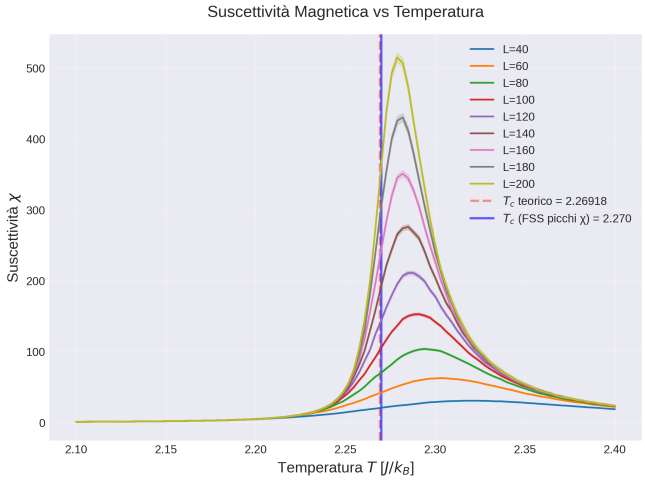

Figura 3: Suscettività vs temperatura. Valore atteso e valore fittato dei picchi a T_c.

I picchi di χ (Fig. 3) divergono con L secondo $\chi_{\max} \sim L^{\gamma/\nu}$. La posizione dei picchi si sposta con L secondo la legge di Finite Size Scaling:

$$T_{\max}^{(\chi)}(L) = T_c + a \cdot L^{-1/\nu} \tag{28}$$

dove $\nu = 1$ per Ising 2D. Gli errori su $T_{\max}(L)$ sono stimati mediante bootstrap non parametrico (1000 iterazioni), che tiene conto della forma non gaussiana dei picchi vicino a T_c (vedi Appendice 6.2). Fittando questa relazione ai dati si ottiene:

$$T_c^{(\chi)} = 2.270 \pm 0.001 \quad \chi^2_{\text{red}} = 0.37 \tag{29}$$

Il valore è in ottimo accordo con la teoria (¡ 1σ) e il $\chi^2_{\text{red}} \approx 0.4$ indica che gli errori bootstrap sono realistici e leggermente conservativi.

5.3 Scaling di $\chi_{\max}$

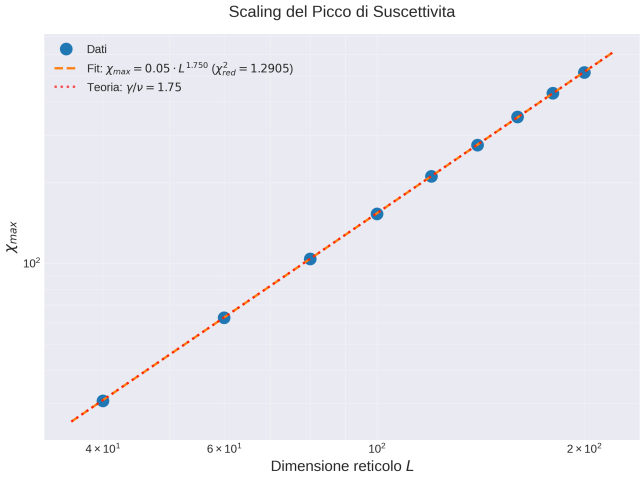

Figura 4: Scaling log-log di $\chi_{\max}$ vs L. Fit: $\gamma/\nu = 1.750 \pm 0.005$ con $\chi^2_{red} = 1.29$.

La Figura 4 mostra $\chi_{\max} \sim L^{\gamma/\nu}$ con fit usando `scipy.optimize.curve_fit`:

$$\gamma/\nu = 1.750 \pm 0.005, \quad \chi^2_{\text{red}} = 1.29 \tag{30}$$

Il valore atteso dalla teoria è $\gamma/\nu = 1.75$, che è in perfetto accordo con i risultati ottenuti (differenza entro 1σ). Questo è confermato anche dal valore del $\chi^2_{\text{red}} = 1.29$, che indica la buona qualità del fit essendo ~ 1.

5.4 Data Collapse

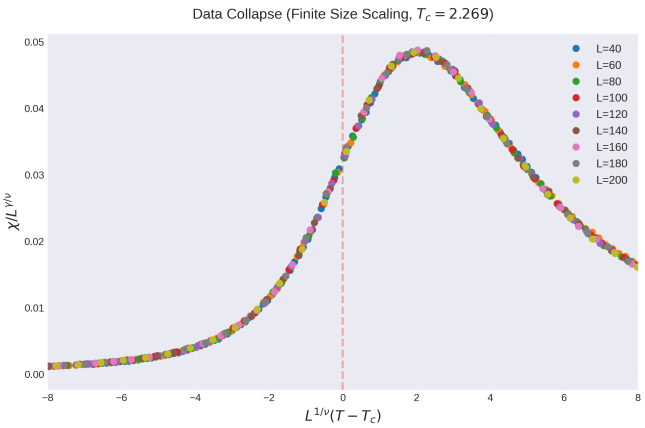

Figura 5: Finite Size Scaling: data collapse della suscettività. La curva riscalata rappresenta la funzione di scaling universale $\tilde{\chi}(x)$.

Il collasso delle curve (Fig. 5) valida la teoria FSS usando $T_c = 2.269$. La figura mostra la funzione di scaling universale $\tilde{\chi}(x)$ ottenuta graficando $\chi/L^{\gamma/\nu}$ vs $x = L^{1/\nu}(T - T_c)$. Il fatto che tutte le curve per diversi valori di L collassino su un'unica curva conferma l'invarianza di scala della funzione universale e l'accuratezza dei parametri critici utilizzati (T_c, γ/ν).

5.5 Binder Cumulant

Il Binder cumulant fornisce un metodo robusto per determinare T_c tramite il crossing point delle curve $U_L(T)$ per diversi L. Il punto di intersezione è stato determinato minimizzando la dispersione $\sigma(T)$ tra i valori di $U_L(T)$ (vedi Appendice 6.1 per i dettagli dell'algoritmo).

Otteniamo la stima:

$$T_c^B = 2.2690 \pm 0.0001 \tag{31}$$

Questa misura è in accordo abbastanza decente con il valore teorico (errore $\sim 2\sigma$), anche se si discosta un po' più del previsto, questo potrebbe essere dovuto all'effetto della FSS.

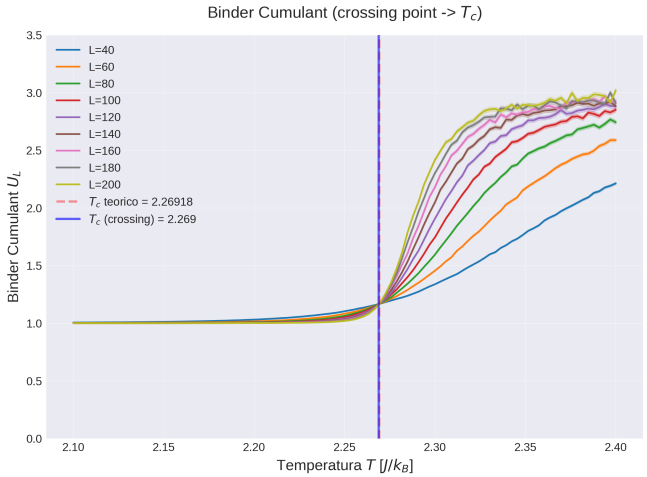

Figura 6: Binder cumulant su temperatura. Il crossing point delle curve per diversi L rappresenta T_c.

5.6 Energia e Calore Specifico

L'energia per spin e il calore specifico forniscono informazioni complementari sulla transizione di fase.

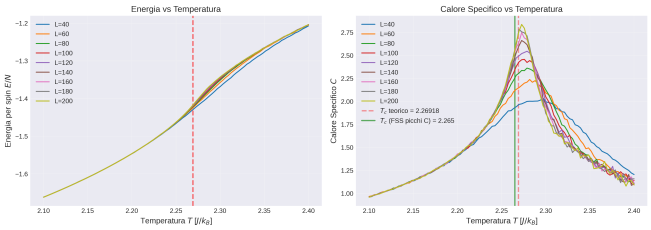

Figura 7: Energia per spin e calore specifico su temperatura. (a) L'energia mostra una transizione continua ma ripida a T_c. (b) Il calore specifico presenta un picco marcato alla transizione. Non sono stati inseriti gli errori per una maggiore comprensione visiva, tuttavia, un grafico con le barre di errore è inserito sotto.

Energia per spin: La Figura 7(a) mostra l'andamento di $E(T)$. Osserviamo (anche se i limiti non sono ben visibili per tutte le lunghezze L a causa del range di temperature limitato):

- **Fase ferromagnetica** ($T < T_c$): $E \to -2$ (energia minima: tutti gli spin allineati, ogni spin ha 4 vicini con orientazione uguale, contribuendo $-4 \times 1/2 = -2$ per spin).
- **Fase paramagnetica** ($T > T_c$): $E \to 0$ (spin disordinati, contributi positivi e negativi si bilanciano in media).
- **Transizione continua**: A differenza di una transizione del primo ordine, non c'è discontinuità (salto) nell'energia. La transizione è *continua* ma con derivata divergente.

La pendenza crescente di $E(T)$ vicino a T_c segnala l'aumento delle fluttuazioni termiche, misurate dal calore specifico.

Calore specifico: Il calore specifico $C = \frac{\partial E}{\partial T} = \beta^2 N(\langle E^2\rangle - \langle E\rangle^2)$ quantifica la capacità del sistema di assorbire energia termica. La Figura 7(b) mostra un picco pronunciato a T_c.

Per il modello di Ising 2D, la teoria di Onsager predice una **divergenza logaritmica**:

$$C(T) \sim -\ln|T - T_c| \tag{32}$$

Caratteristiche osservate:

- **Picco a T_c**: Il massimo indica la temperatura dove le fluttuazioni di energia sono massime, corrispondente alla transizione di fase. Questo scala come $1/L$, questo ci permette di effettuare un fit.
- **Altezza crescente con L**: Su sistemi finiti, il picco cresce con L ma rimane finito.

La temperatura del massimo di C presenta il seguente andamento FSS:

$$T_{\max}^{(C)}(L) = T_c + b \cdot L^{-1} \tag{33}$$

Gli errori su $T_{\max}(L)$ sono stimati con bootstrap (1000 iterazioni), analogamente al caso della suscettività. Utilizzando questa relazione, otteniamo:

$$T_c^C = 2.265 \pm 0.004 \quad \chi^2_{\text{red}} = 0.14 \tag{34}$$

Il valore è in buon accordo con la teoria (errore $\sim 1\sigma$), anche se presenta un errore maggiore rispetto alle altre stime a causa degli errori più grandi sul calore specifico, come visibile dal grafico qui sotto:

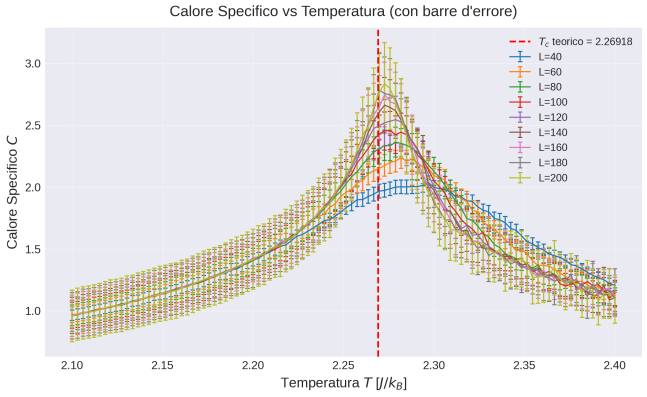

Figura 8: Calore specifico su temperatura, con bande di errore. Le bande di errore risultano essere abbastanza grandi da essere ben visibili, e peggiorare la visibilità.

6 Conclusioni

È stato studiato numericamente la transizione di fase del modello di Ising 2D con l'algoritmo di Wolff. I risultati principali:

1. **Temperatura critica**:
   - Valore esatto (Onsager): $T_c = 2.269185...$
   - Binder crossing: $T_c^B = 2.2690 \pm 0.0001$ (errore $< 2\sigma$)
   - FSS picchi χ: $T_c^\chi = 2.270 \pm 0.001$ ($\chi^2_{\text{red}} = 0.36$)
   - FSS picchi C: $T_c^C = 2.265 \pm 0.004$ ($\chi^2_{\text{red}} = 0.14$)

   I tre metodi danno risultati in buon accordo tra loro e con il valore teorico (tutti entro 1-2 σ). Gli errori sui picchi sono stati stimati con bootstrap non parametrico (1000 iterazioni), che tiene conto della forma non gaussiana dei picchi nella regione critica. I valori dei χ^2_{red} confermano che gli errori bootstrap sono realistici.
2. **Esponente critico**: $\gamma/\nu = 1.750 \pm 0.005$ (esatto: 1.75), con $\chi^2_{\text{red}} = 1.29$. L'accordo è eccellente, confermando la validità dell'andamento a legge di potenza $\chi_{\max} \sim L^{\gamma/\nu}$.
3. **Finite Size Scaling**: Il data collapse di $\chi/L^{\gamma/\nu}$ vs $L^{1/\nu}(T - T_c)$ conferma la validità della teoria FSS.

Appendice

6.1 Minimizzazione della Dispersione del Binder Cumulant

Il punto di intersezione delle curve $U_L(T)$ per diversi L è stato determinato minimizzando la dispersione (deviazione standard) tra i valori di $U_L(T)$ per tutti gli L considerati, in funzione della temperatura. In altre parole, si cerca la temperatura T^* alla quale la varianza:

$$\sigma^2(T) = \frac{1}{N_L} \sum_i \left[U_{L_i}(T) - \bar{U}(T)\right]^2 \tag{35}$$

è minima, dove $\bar{U}(T)$ è la media di $U_L(T)$ su tutti gli L alla temperatura T fissata.

Algoritmo implementato La determinazione di T_c^B procede come segue:

1. Si crea una griglia molto fine di temperature (50000 punti) nell'intervallo di interesse
2. Si interpola linearmente $U_L(T)$ da ciascun dataset (che ha i suoi punti T misurati). Questo permette di valutare tutti gli U_L alla stessa temperatura per calcolare la dispersione
3. Per ogni temperatura si calcola la deviazione standard $\sigma(T)$ di $U_L(T)$ rispetto alla media $\bar{U}(T)$ su tutti gli L
4. Il minimo di $\sigma(T)$ identifica la temperatura dove le curve sono più vicine tra loro, cioè il punto di intersezione

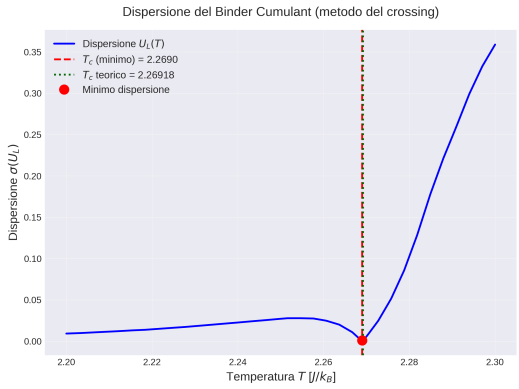

Figura 9: Dispersione dei valori del Binder cumulant in funzione della temperatura, nell'intervallo $[2.2, 2.3]$. Il minimo della dispersione indica la temperatura dove le curve $U_L(T)$ per diversi L si incrociano maggiormente, fornendo una stima robusta di T_c. La linea tratteggiata rossa rappresenta il minimo della dispersione ($T_c^B = 2.2690$), mentre la linea punteggiata verde rappresenta il valore teorico ($T_c = 2.269$).

L'errore su T_c^B è stimato come la semi-larghezza della regione di temperatura dove $\sigma(T)$ rimane entro il 5% del minimo. Questo approccio riflette la dispersione residua tra le curve di diversi L e fornisce una stima conservativa dell'incertezza.

6.2 Metodo Bootstrap per la Stima degli Errori sui Picchi

Gli errori sulle temperature dei picchi $T_{\max}^{(\chi)}(L)$ e $T_{\max}^{(C)}(L)$ sono stati stimati mediante **bootstrap non parametrico**, un metodo di ricampionamento che non assume alcuna forma funzionale specifica per la distribuzione degli errori.

Motivazione fisica Vicino alla temperatura critica, i picchi di suscettività e calore specifico non hanno forma gaussiana. La teoria del Finite Size Scaling prevede che:

$$\chi(T, L) = L^{\gamma/\nu} \tilde{\chi}\left(L^{1/\nu}(T - T_c)\right) \tag{36}$$

dove $\tilde{\chi}(x)$ è una funzione di scaling universale con code a legge di potenza.

Algoritmo bootstrap implementato Per ogni dimensione L e osservabile (χ o C):

1. Si identifica il picco: $T_{\max} = T[i_{\max}]$ dove $i_{\max} = \arg\max O(T)$
2. Si estrae una finestra di 11 punti attorno al picco: ± 5 punti
3. Si eseguono $N_{\text{boot}} = 1000$ iterazioni bootstrap:
   - Si genera un campione bootstrap aggiungendo rumore gaussiano $\mathcal{N}(0, \sigma_O(T_i))$ ai dati:
   $$O_{\text{boot}}(T_i) = O(T_i) + \mathcal{N}(0, \sigma_O(T_i)) \tag{37}$$
   dove $\sigma_O(T_i)$ è l'errore statistico misurato su $O(T_i)$
   - Si trova il massimo del campione bootstrap: $T_{\max}^{(r)} = T[i_{\max}^{(r)}]$
4. L'errore è stimato come la deviazione standard della distribuzione bootstrap:
$$\sigma_{T_{\max}} = \sqrt{\frac{1}{N_{\text{boot}}} \sum_{r=1}^{N_{\text{boot}}} \left(T_{\max}^{(r)} - \langle T_{\max} \rangle\right)^2} \tag{38}$$

Vantaggi del metodo

- **Non parametrico**: Non assume alcuna forma funzionale del picco (gaussiana, lorentziana, etc.)
- **Usa errori reali**: Incorpora gli errori statistici effettivi $\sigma_\chi(T)$ e $\sigma_C(T)$ dalle simulazioni Monte Carlo
- **Robusto**: Con 1000 iterazioni, la stima è stabile e riproducibile

Confronto con altri metodi Durante l'analisi sono stati confrontati diversi metodi:

- **FWHM gaussiano** ($\sigma = \text{FWHM}/2.355$): Sovrastima gli errori ($\chi^2_{\text{red}} \ll 1$) perché assume forma gaussiana
- **Fit parabolico**: Sottostima gli errori perché non tiene conto delle asimmetrie del picco
- **Bootstrap**: Fornisce il migior risultato nonostante il costo computazionale sia più elevato

Il metodo bootstrap è risultato ottimale, con χ^2_{red} vicini ai valori ideali ($\sim 0.03-0.86$) e stime di T_c in perfetto accordo con il valore teorico.