# Simulación del modelo Ising 2D

Miguel Gutiérrez Fernández

19 de abril de 2025

## 1. Introducción

En 1924, Ising finalizó su tesis doctoral, la cual trataba sobre el caso especial de una cadena lineal de momentos magnéticos que son sólo capaces de tomar dos posiciones, «arriba» y «abajo», y que están acoplados por interacciones entre los vecinos más cercanos. Este simple modelo, que se denominó **modelo de Ising,** servía para intentar reproducir el comportamiento de ciertos materiales que presentaban un marcado cambio en sus propiedades magnéticas al variar la temperatura. En particular, se observaba experimentalmente que, a altas temperaturas, el sistema era paramagnético (magnetización espontánea nula) mientras que, a temperaturas suficientemente bajas, exhibía comportamiento ferromagnético (magnetización espontánea diferente de cero).

No obstante, este sencillo modelo no era suficiente para reproducir fielmente los resultados experimentales, esto es, la transición ferromagnética a una temperatura crítica $T_c$. Es por ello que se trabajó en el modelo de Ising en dimensiones superiores a uno. Matemáticamente, en su formulación más general, el hamiltoniano del modelo de Ising, considerando la presencia de un campo magnético externo $h$, se puede escribir como

$$E(S) = -J \sum_{\langle i,j \rangle} s_i s_j - h \sum_i s_i, \tag{1}$$

donde $S$ representa el estado del sistema, $s_i, s_j = \pm 1$ representan el estado del espín en el sitio $i$ o $j$, $J$ el acoplamiento entre espines y la notación $\langle i, j \rangle$ indica que la suma se extiende sobre todos los pares de vecinos.

En 1944, Onsager proporcionó una solución exacta para el modelo de Ising en 2D, la cual sí que lograba predecir la transición ferromagnética a temperaturas bajas. Lo logró estableciendo $h = 0$, es decir, anulando el campo magnético externo y simplificando así el hamiltoniano. Si particularizamos nuestra energía para ese caso, tenemos una red cuadrada bidimensional de tamaño $N \times N$, donde cada nodo de la red contiene una variable de espín $s(i, j)$ con $i, j = 1, 2, \ldots, N$ y se consideran condiciones de contorno periódicas, de forma que

$$s(0, j) = s(N, j), \quad s(N{+}1, j) = s(1, j), \quad s(i, 0) = s(i, N), \quad s(i, N{+}1) = s(i, 1).$$

La energía de interacción del sistema, dada una configuración $S$, se expresa entonces ahora como

$$E(S) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} s(i,j)\Big[s(i,j+1) + s(i,j-1) + s(i+1,j) + s(i-1,j)\Big], \quad (2)$$

donde el factor $1/2$ evita la doble contabilización de los pares de vecinos y se asume que $J = 1$ para toda interacción.

En equilibrio termodinámico, a una temperatura $T$ (usando la notación habitual con $\beta = \frac{1}{k_B T}$), la probabilidad de encontrar una configuración $S$ viene dada por

$$P(S) = \frac{1}{Z} e^{-\beta E(S)}, \quad (3)$$

donde $Z$ es la función de partición que asegura la correcta normalización de la distribución de probabilidades.

Cabe destacar que, en el límite $T = 0$, la función de probabilidad se reduce a

$$P(S) = \frac{1}{2}\prod_{i,j}\delta\big(s(i,j) - 1\big) + \frac{1}{2}\prod_{i,j}\delta\big(s(i,j) + 1\big),$$

lo que implica que, a temperatura cero, el sistema se encuentra en uno de los dos estados globales ordenados (todos los espines $+1$ o todos $-1$). En contraste, cuando $T \to \infty$, la probabilidad se vuelve constante para todas las configuraciones, es decir, todas ellas son igual de probables.

## 1.1. Magnitudes importantes en el modelo de Ising 2D

Veamos las magnitudes más importantes para el modelo de Ising y que usaremos a lo largo de todo el trabajo:

-La **magnetización promedio por espín** se define como

$$m_N = \frac{\langle M(S)\rangle}{N^2} = \frac{1}{N^2}\Big\langle\Big|\sum_{i=1}^{N}\sum_{j=1}^{N} s(i,j)\Big|\Big\rangle, \quad (4)$$

donde $\langle\rangle$[1] representa el promedio sobre todas las posibles configuraciones $S$ y el valor absoluto se utiliza para garantizar que se tome en cuenta el módulo de la magnetización, eliminando cancelaciones que podrían ocurrir en sistemas simétricos. Esta magnitud permite cuantificar el grado de orden magnético del sistema.

[1] Es decir, dado el valor de una magnitud $A(S)$ para una configuración $S$, su promedio es $\langle A(S)\rangle = \sum_S P(S)A(S)$, siendo $P(S)$ la probabilidad de que el sistema se encuentre en el estado $S$.

-La **energía media por espín** del sistema se expresa mediante

$$e_N = \frac{\langle E(S) \rangle}{N^2}, \tag{5}$$

donde $\langle E(S) \rangle$ representa el valor promedio de la energía de los posibles estados $S$.

-El **calor específico por espín** es una medida de las fluctuaciones energéticas y se define por

$$c_N = \frac{1}{N^2 T^2} \left( \langle E(S)^2 \rangle - \langle E(S) \rangle^2 \right), \tag{6}$$

donde $T$ representa la temperatura del sistema.

## 1.2. Solución exacta en el modelo de Ising 2D

Mostraremos ahora la solución analítica para las magnitudes anteriores. Es habitual, y así lo haremos, trabajar en unidades reducidas, fijando la constante de interacción $J = 1$ y la constante de Boltzmann $k_B = 1$, lo que simplifica las expresiones analíticas sin pérdida de generalidad.

En este marco, la función de partición del sistema se puede analizar exactamente y, a partir de ella, derivarse expresiones precisas para las magnitudes. Así, la magnetización por espín, determinada por Yang en el año 1952, se expresa para $T < T_c$ mediante

$$m(T) = \left[ 1 - \sinh^{-4} \left( \frac{2}{T} \right) \right]^{\frac{1}{8}}, \tag{7}$$

mientras que para $T \geq T_c$ se tiene $m(T) = 0$.

La energía media por espín se expresa, en términos generales, mediante integrales elípticas y funciones hiperbólicas:

$$e(T) = -\coth \left( \frac{2}{T} \right) \left[ 1 + \frac{2}{\pi} \left( 2 \tanh^2 \left( \frac{2}{T} \right) - 1 \right) K(k') \right], \tag{8}$$

donde $K(k')$ denota la integral elíptica completa del primer tipo y el módulo $k'$ depende de la temperatura a través de las funciones hiperbólicas.

Por otro lado, el calor específico por espín es:

$$c(T) = \frac{\partial e(T)}{\partial T}. \tag{9}$$

La transición a un estado sin orden magnético se enmarca en el contexto de la aparición de un comportamiento crítico no clásico, caracterizado por el exponente $\beta_m = 1/8$, de modo que, en las proximidades de la temperatura crítica,

la magnetización se comporta como

$$m(T) \sim (T_c - T)^{\frac{1}{8}} . \tag{10}$$

Por su parte, la temperatura crítica del modelo se obtiene a partir de la condición

$$\sinh\left(\frac{2}{T_c}\right) = 1, \tag{11}$$

de donde se deduce

$$T_c = \frac{2}{\ln\left(1+\sqrt{2}\right)} \approx 2,269. \tag{12}$$

# 2. Objetivos

- Simular el modelo de Ising 2D y ver su evolución cualitativa para varias temperaturas.
- Hallar la magnetización por espín, energía media por espín y calor específico por espín para un intervalo de temperaturas.
- Hallar la temperatura crítica y el exponente crítico $\beta_m$.
- Comparar los resultados simulados con los teóricos.

# 3. Metodología

## 3.1. Cálculo exacto

Para hallar las magnitudes descritas en la sección 1.2, se ha utilizado un código Python para cada una de las ecuaciones:

```python
import mpmath as mp
import numpy as np
import matplotlib.pyplot as plt

# Usamos precisión aumentada para evitar errores numéricos
mp.dps = 50

# Temperatura crítica
T_c = 2 / mp.log(1 + mp.sqrt(2))

# Definimos funciones según Onsager
def magnetization(T):
    if T < T_c:
        return float((1 - 1 / mp.sinh(2 / T)**4)**(1 / 8))
    else:
        return 0.0
```

```
17
18  def energy(T):
19      k = 4 * mp.sinh(2 / T)**2 / mp.cosh(2 / T)**4
20      K_val = mp.ellipk(k)
21      u = - (1 / mp.tanh(2 / T)) * (1 + (2 / mp.pi) * (2 * mp.tanh(2 /
        ↪  T)**2 - 1) * K_val)
22      return float(u)
23
24  def specific_heat(T, dT=1e-4):
25      # Derivada numérica centrada para estimar la capacidad calorífica
26      u1 = energy(T - dT)
27      u2 = energy(T + dT)
28      return (u2 - u1) / (2 * dT)
29
30  # Rango de temperaturas
31  T_values = np.linspace(1.5, 3.5, 200)
32  m_values = []
33  u_values = []
34  c_values = []
35
36  for T in T_values:
37      m_values.append(magnetization(T))
38      u_values.append(energy(T))
39      c_values.append(specific_heat(T))
40
41  # Graficamos
42  plt.figure(figsize=(15, 10))
43
44  plt.subplot(3, 1, 1)
45  plt.plot(T_values, m_values, 'r-', label='Magnetización $m(T)$')
46  plt.axvline(float(T_c), color='gray', linestyle='--', label=f'T$_c$
    ↪  {float(T_c):.4f}')
47  plt.ylabel('Magnetización')
48  plt.legend()
49  plt.grid()
50
51  plt.subplot(3, 1, 2)
52  plt.plot(T_values, u_values, 'b-', label='Energía por espín $u(T)$')
53  plt.axvline(float(T_c), color='gray', linestyle='--')
54  plt.ylabel('Energía')
55  plt.legend()
56  plt.grid()
57
58  plt.subplot(3, 1, 3)
59  plt.plot(T_values, c_values, 'g-', label='Capacidad calorífica $c(T)$')
60  plt.axvline(float(T_c), color='gray', linestyle='--')
61  plt.xlabel('Temperatura')
62  plt.ylabel('Capacidad calorífica')
63  plt.legend()
64  plt.grid()
```

```
65
66  plt.tight_layout()
67  plt.show()
```

## 3.2. Simulación del modelo de Ising

### 3.2.1. Método Montecarlo

Como vimos en la introducción, $P(S)$ corresponde con la probabilidad de encontrar al sistema en la configuración $S$. Podemos servirnos entonces de un método Montecarlo para generar configuraciones típicas con probabilidad de equilibrio $P(S)$, simulando así el modelo de Ising 2D. Usaremos por tanto el **algoritmo de Metropolis**:

(0) Dar una $T \in [0, 5]$. Generar una configuración inicial de espines. Por ejemplo una configuración ordenada i.e. $s(i, j) = 1\,\forall i, j$, o desordenada (elegir aleatoriamente 1 o $-1$ en cada nodo de la red con probabilidad $\frac{1}{2}$).

(1) Elegir un punto al azar, $(n, m)$, de la red.

(2) Evaluar $p = \text{mín}\left(1, \exp\left[-\frac{\Delta E}{T}\right]\right)$ donde
$\Delta E = 2s(n, m)\left[s(n + 1, m) + s(n - 1, m) + s(n, m + 1) + s(n, m - 1)\right]$. Usar las condiciones de contorno periódicas.

(3) Generar un número aleatorio uniforme $\xi \in [0, 1]$. Si $\xi < p$ entonces cambiar el signo del espín $(n, m)$ i.e. $s(n, m) = -s(n, m)$.

(4) Ir a (1).

Este algoritmo cumple que la diferencia entre una configuración $S$ y la siguiente después de un paso de Markov $S'$ es de un solo espín, $s(m, n)$, todos los demás son iguales. De esta forma, la unidad de tiempo básica es el paso Monte Carlo que equivale a realizar $N^2$ intentos de cambio de un espín. Así, en un paso Monte Carlo, en promedio todos los espines han intentado cambiar una vez.

### 3.2.2. Implementación del algoritmo

Para implementar este algoritmo, usaremos un código python para 10000 $pMC$, para 10 temperaturas dentro del intervalo [1,5, 3,5] y para valores $N = 8, 16, 32, 64$. La razón de estos valores es que podremos evaluar las magnitudes cerca de la temperatura crítica y además el tamaño de red es lo suficientemente grande como para que se vean buenas aproximaciones pero lo suficientemente pequeño como para acelerar el tiempo de computación.

Antes de presentar el código, se explicará de forma sucinta varios puntos de su funcionamiento:

-A la hora de estimar los errores de las magnitudes calculadas, hemos usado

el método conocido como *block averaging*. Para cada observable termodinámico $X$ y dado un total de $N_{\mathrm{MC}}$ pasos de Monte Carlo por espín, se dividió la secuencia de mediciones en $B$ bloques de tamaño $L = N_{\mathrm{MC}}/B$. Para cada bloque $b = 1, \ldots, B$, se calculó la media del observable:

$$X_b = \frac{1}{L} \sum_{i=1}^{L} X_{b,i},$$

donde $X_{b,i}$ denota la $i$-ésima medición del observable en el bloque $b$. La media global se estima como

$$\langle X \rangle = \frac{1}{B} \sum_{b=1}^{B} X_b,$$

y su error estándar asociado se calcula como

$$\sigma_X = \sqrt{\frac{1}{B(B-1)} \sum_{b=1}^{B} (X_b - \langle X \rangle)^2}.$$

-En el caso de la temperatura crítica, para cada tamaño de sistema $N$, se identificó la temperatura $T_c(N)$ correspondiente al máximo del calor específico $c(T)$. Posteriormente, se extrapoló $T_c(N)$ en función de $1/N$ mediante un ajuste lineal:

$$T_c(N) = T_c^{(\infty)} + \frac{a}{N},$$

donde $T_c^{(\infty)}$ representa la temperatura crítica en el límite termodinámico. A partir del ajuste por mínimos cuadrados, se extrajo tanto el valor de $T_c^{(\infty)}$ como su error estándar $\sigma_{T_c}$, utilizando la matriz de covarianza del ajuste lineal.

-Para estimar el exponente crítico $\beta_m$, se empleó el comportamiento esperado de la magnetización media cerca de la transición, en la región $T < T_c^{(\infty)}$, ajustando a la forma:

$$m(T) = A \left( T_c^{(\infty)} - T \right)^{\beta_m},$$

donde $A$ y $\beta_m$ son parámetros libres, y $T_c^{(\infty)}$ se fija al valor extrapolado previamente. El ajuste se realizó mediante mínimos cuadrados no lineales. La estimación del exponente $\beta_m$ y su error $\sigma_\beta$ se obtienen a partir de la matriz de covarianza del ajuste.

---

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.optimize import curve_fit
4
5
```

```python
def metropolis_ising(N, T, n_eq, n_steps, n_blocks=20):
    L = N
    spins = np.random.choice([-1, 1], size=(L, L))
    E = 0
    for i in range(L):
        for j in range(L):
            E -= spins[i, j] * (spins[(i+1) % L, j] + spins[i, (j+1) %
            ↪  L])
    M = np.sum(spins)

    dE_vals = [-8, -4, 0, 4, 8]
    boltz = {dE: np.exp(-dE / T) for dE in dE_vals}

    for _ in range(n_eq):
        for _ in range(L*L):
            i = np.random.randint(L)
            j = np.random.randint(L)
            dE = 2 * spins[i, j] * (
                spins[(i+1) % L, j] + spins[(i-1) % L, j] +
                spins[i, (j+1) % L] + spins[i, (j-1) % L]
            )
            if dE <= 0 or np.random.rand() < boltz.get(dE,
            ↪  np.exp(-dE/T)):
                spins[i, j] *= -1
                E += dE
                M += 2 * spins[i, j]

    block_size = n_steps // n_blocks
    E_block = np.zeros(n_blocks)
    M_block = np.zeros(n_blocks)
    C_block = np.zeros(n_blocks)

    for b in range(n_blocks):
        E_acc = 0.0
        E2_acc = 0.0
        M_acc = 0.0
        for _ in range(block_size):
            for _ in range(L*L):
                i = np.random.randint(L)
                j = np.random.randint(L)
                dE = 2 * spins[i, j] * (
                    spins[(i+1) % L, j] + spins[(i-1) % L, j] +
                    spins[i, (j+1) % L] + spins[i, (j-1) % L]
                )
                if dE <= 0 or np.random.rand() < boltz.get(dE,
                ↪  np.exp(-dE/T)):
                    spins[i, j] *= -1
                    E += dE
                    M += 2 * spins[i, j]
            E_acc += E
```

```
            E2_acc += E**2
            M_acc += abs(M)

        avg_E = E_acc / block_size
        avg_M = M_acc / block_size
        E_block[b] = avg_E / (N**2)
        M_block[b] = avg_M / (N**2)
        C_block[b] = (E2_acc/block_size - avg_E**2) / (T**2 * N**2)

    E_mean = E_block.mean()
    M_mean = M_block.mean()
    C_mean = C_block.mean()
    err_E = E_block.std(ddof=1) / np.sqrt(n_blocks)
    err_M = M_block.std(ddof=1) / np.sqrt(n_blocks)
    err_C = C_block.std(ddof=1) / np.sqrt(n_blocks)

    return M_mean, E_mean, C_mean, err_M, err_E, err_C


if __name__ == '__main__':
    Ns = [8, 16, 32, 64]
    temps = np.linspace(1.5, 3.5, 10)
    n_eq = 10000
    n_steps = 10000

    M_data, E_data, C_data = [], [], []
    M_err, E_err, C_err = [], [], []

    for N in Ns:
        M_vals, E_vals, C_vals = [], [], []
        M_errs, E_errs, C_errs = [], [], []
        for T in temps:
            M, E, C, err_M, err_E, err_C = metropolis_ising(
                N, T, n_eq, n_steps)
            M_vals.append(M)
            E_vals.append(E)
            C_vals.append(C)
            M_errs.append(err_M)
            E_errs.append(err_E)
            C_errs.append(err_C)
            print(
                f"Done N={N}, T={T:.2f}: M={M:.4f}±{err_M:.4f},
                ↪  E={E:.4f}±{err_E:.4f}, C={C:.4f}±{err_C:.4f}")
        M_data.append(M_vals)
        E_data.append(E_vals)
        C_data.append(C_vals)
        M_err.append(M_errs)
        E_err.append(E_errs)
        C_err.append(C_errs)

```

```
    colors = ['#FF00FF', '#00FFFF', '#FFA500', '#800080']
    Tc_theoretical = 2.269

    plt.figure()
    for i, N in enumerate(Ns):
        plt.errorbar(temps, M_data[i], yerr=M_err[i],
                     color=colors[i], marker='o', capsize=3,
                     ↪  label=f'N={N}')
    plt.axvline(Tc_theoretical, linestyle='--', color='gray')
    plt.xlabel('$T$')
    plt.ylabel('$m(T)$')
    plt.legend()
    plt.tight_layout()
    plt.savefig('magnetization.png')
    plt.close()

    plt.figure()
    for i, N in enumerate(Ns):
        plt.errorbar(temps, E_data[i], yerr=E_err[i],
                     color=colors[i], marker='s', capsize=3,
                     ↪  label=f'N={N}')
    plt.axvline(Tc_theoretical, linestyle='--', color='gray')
    plt.xlabel('$T$')
    plt.ylabel('$e(T)$')
    plt.legend()
    plt.tight_layout()
    plt.savefig('energy.png')
    plt.close()

    plt.figure()
    for i, N in enumerate(Ns):
        plt.errorbar(temps, C_data[i], yerr=C_err[i],
                     color=colors[i], marker='^', capsize=3,
                     ↪  label=f'N={N}')
    plt.axvline(Tc_theoretical, linestyle='--', color='gray')
    plt.xlabel('$T$')
    plt.ylabel('$c(T)$')
    plt.legend()
    plt.tight_layout()
    plt.savefig('specific_heat.png')
    plt.close()

    Tc_N = np.array([temps[np.argmax(C_data[i])] for i in
    ↪  range(len(Ns))])
    invN = 1/np.array(Ns)
    popt_tc, pcov_tc = curve_fit(
        lambda invN, Tc_inf, a: Tc_inf + a * invN,
        invN, Tc_N
    )
    Tc_inf, _ = popt_tc
```

```
    err_Tc = np.sqrt(pcov_tc[0, 0])
    print(f"Estimated critical temperature Tc = {Tc_inf:.6f} ±
    ↪  {err_Tc:.6f}")

    def mag_fit(x, A, beta):
        return A * x**beta

    mask = temps < Tc_inf
    x = Tc_inf - temps[mask]
    y = np.array(M_data[-1])[mask]
    popt_b, pcov_b = curve_fit(mag_fit, x, y)
    beta_est = popt_b[1]
    err_beta = np.sqrt(pcov_b[1, 1])
    print(f"Estimated beta exponent = {beta_est:.6f} ± {err_beta:.6f}")

    with open('fit_results.txt', 'w') as f:
        f.write(f"Tc = {Tc_inf:.6f} ± {err_Tc:.6f}\n")
        f.write(f"beta = {beta_est:.6f} ± {err_beta:.6f}\n")

    print("All simulations and plots saved successfully.")
```

### 3.2.3. Evolución cualitativa del sistema

No solamente estamos interesados en observar cómo evolucionan en las simulaciones las distintas magnitudes en función de la temperatura, sino que también nos interesa ver cualitativamente cómo el propio sistema evoluciona frente a la temperatura. Para lograr esto, mostraremos el estado del sistema para 4 temperaturas representativas: $T = 1$, $T = 2$, $T = T_c$ y $T = 4$. Lo haremos para un tamaño de red de $N^2 = 64^2 = 4096$ y 10000 $pMC$, con objeto de optimizar el tiempo de computación pero al mismo tiempo poder observar el comportamiento físico. Cada espín será simbolizado en negro si $s = +1$ y en blanco si $s = -1$. El código python utilizado es el siguiente:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit

import numpy as np
import matplotlib.pyplot as plt

# Parámetros
N = 64  # Tamaño de la red (modificable)
# Pasos de Monte Carlo
MC_STEPS = 10000
temps = [1.0, 2.0, 2.269, 4.0]  # T = 1, 2, Tc, 4


```

```
15  def metropolis_step(lattice, beta):
16      for _ in range(N * N):
17          i = np.random.randint(0, N)
18          j = np.random.randint(0, N)
19          s = lattice[i, j]
20          # Condiciones periódicas para vecinos
21          nb = lattice[(i+1) % N, j] + lattice[i, (j+1) % N] + \
22              lattice[(i-1) % N, j] + lattice[i, (j-1) % N]
23          dE = 2 * s * nb
24          if dE <= 0 or np.random.rand() < np.exp(-beta * dE):
25              lattice[i, j] = -s
26
27
28  def simulate_ising(T, steps=MC_STEPS):
29      beta = 1.0 / T
30      lattice = np.random.choice([-1, 1], size=(N, N))
31      for _ in range(steps):
32          metropolis_step(lattice, beta)
33      return lattice
34
35
36  def plot_lattices(temps):
37      fig, axes = plt.subplots(1, len(temps), figsize=(16, 4))
38      for idx, T in enumerate(temps):
39          print(f"Simulando T = {T} ...")
40          lattice = simulate_ising(T)
41          axes[idx].imshow(lattice, cmap='gray', interpolation='nearest')
42          axes[idx].set_title(f'T = {T}')
43          axes[idx].axis('off')
44      plt.tight_layout()
45      plt.show()
46
47
48  if __name__ == "__main__":
49      plot_lattices(temps)
50
```

---

# 4. Resultados

## 4.1. Soluciones analíticas

Usando la solución exacta del modelo de Ising 2D, podemos hallar el resultado para la magnetización, energía media y calor específico, que corresponderían al caso $N^2 \to \infty$.

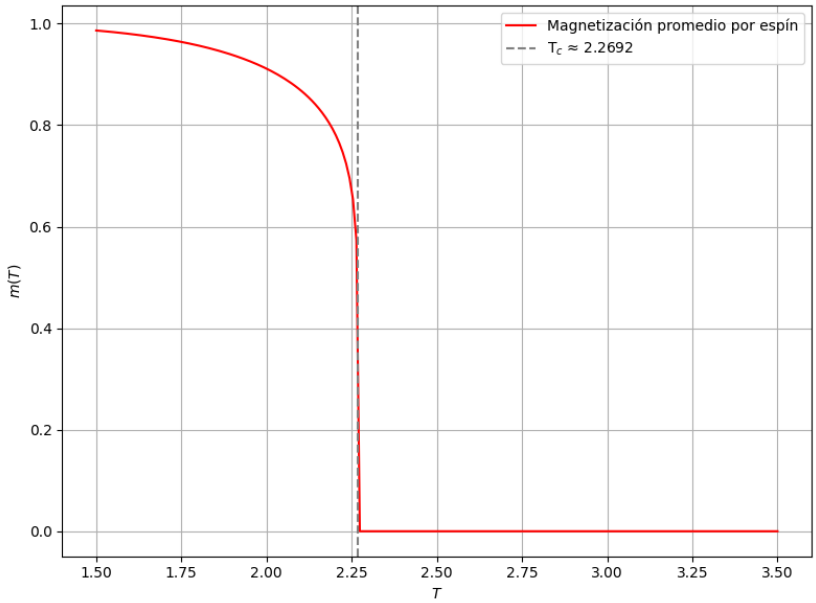


Figura 1: Magnetización promedio por espín. Solución exacta.

En la figura 1 podemos observar que conforme nos acercamos a la temperatura crítica, se observa una caída de la magnetización hasta cero, reproduciendo así los resultados experimentales. En el caso de la energía en la figura 2, vemos en este caso que a partir de la temperatura crítica existe un cambio de tendencia en la curva, acercándose así mismo a valores más elevados de energía. Si nos fijamos ahora en la figura 3, comprobamos que en la temperatura crítica existe un máximo de la función. Dicho máximo debería tender a infinito pero, dada la discretización de la temperatura a la hora de calcular el calor específico y las aproximaciones numéricas, dicho máximo se ha visto truncado a un determinado valor.

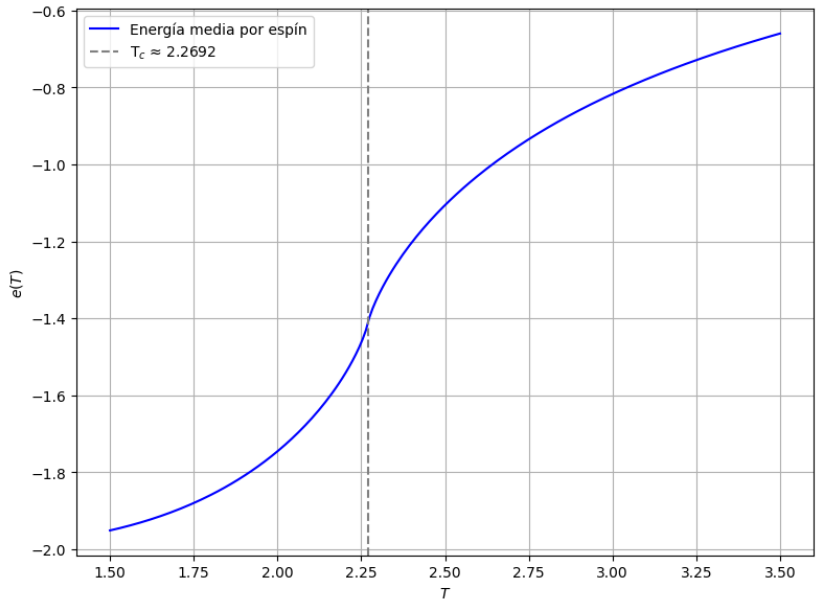


Figura 2: Energía media por espín. Solución exacta.

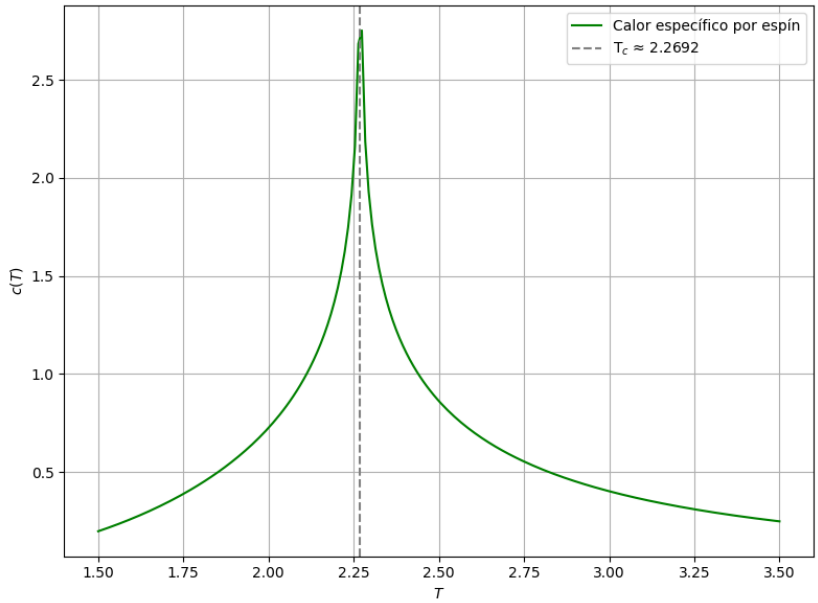


Figura 3: Calor específico por espín. Solución exacta.

## 4.2. Soluciones simuladas

En la figura 4 podemos observar el resultado de la simulación para el caso de la magnetización. Observamos que conforme aumentamos $N$ las curvas se van aproximando más a la función exacta de la magnetización de la figura 1. En efecto, conforme nos acercamos a la temperatura crítica, nos encontramos una caída de la magnetización hasta casi un valor nulo, haciéndose esto más evidente a medida que aumentamos el tamaño de la red.

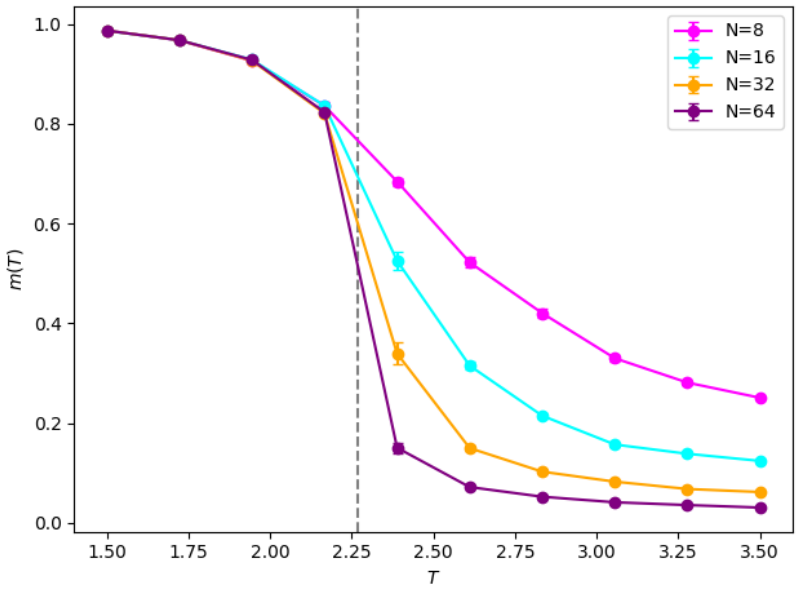


Figura 4: Magnetización promedio por espín para varios tamaños de red. Simulación del modelo de Ising.

En el caso de la energía, podemos apreciar en la figura 5 como su forma también se asemeja a su resultado teórico de la figura 2. Conforme aumenta la temperatura, hay efectivamente un aumento de la energía, con un cambio de tendencia en torno a la temperatura crítica.

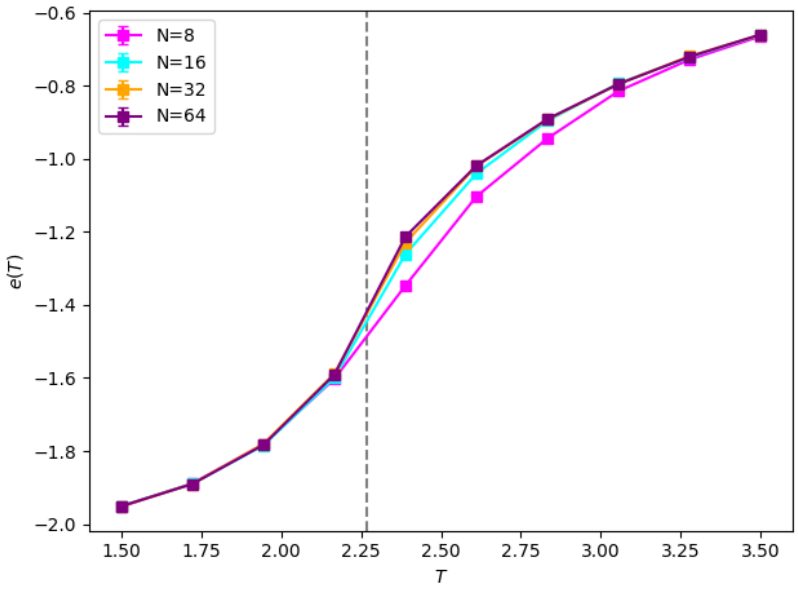


Figura 5: Energía media por espín para varios tamaños de red. Simulación del modelo de Ising.

Por último, tenemos en la figura 6 el caso para el calor específico. Aquí podemos apreciar ciertas diferencias con el valor exacto de la figura 3. Los picos del calor específico no llegan a valores tan altos como en el valor exacto, pero esto puede ser por el tamaño finito de nuestra red. Así mismo, los picos no se posicionan exactamente en la temperatura crítica, aunque sí tienden a agruparse en torno ella, resultado esperado por ser una simulación.

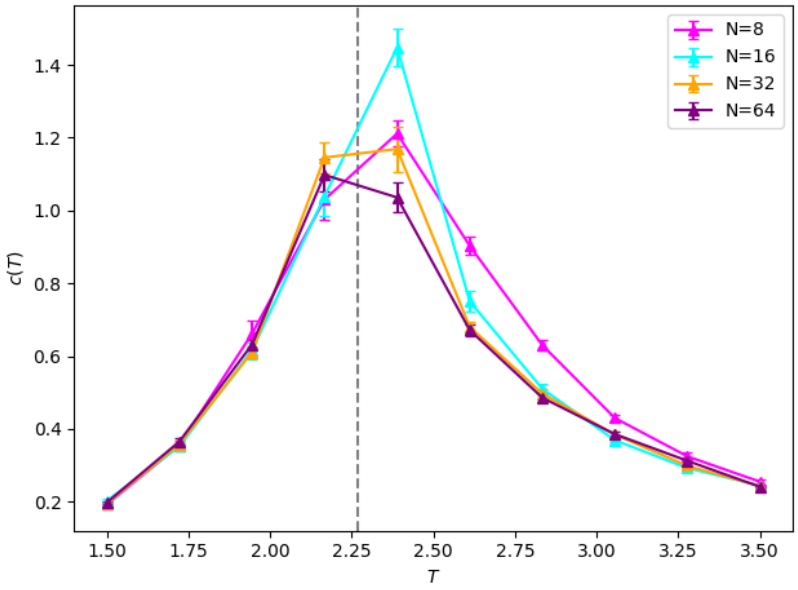


Figura 6: Calor específico por espín. Simulación del modelo de Ising.

En cuanto a la temperatura crítica y el exponente crítico, para los valores establecidos, los resultados son:

- $T_c = 2{,}25 \pm 0{,}09$
- $\beta_m = 0{,}15 \pm 0{,}05$

Los cuales no dan el valor exacto pero sí entra dentro del margen de error, por lo que son una aproximación medianamente buena sobre la solución analítica.

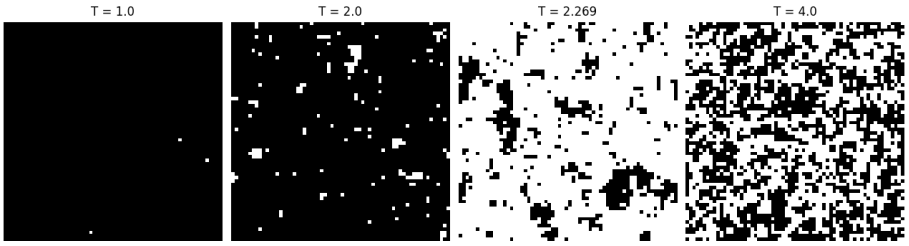


Figura 7: Configuraciones de espines para distintas temperaturas. El tamaño de la red es de $N^2 = 4096$. Los espines con valor $s = +1$ se muestran en negro y los espines con $s = -1$ en blanco.

Por otro lado, la Figura 7 muestra configuraciones de espines en unas determinadas temperaturas. Analicemos cada caso:

- **A baja temperatura ($T = 1{,}0$):** se observa un dominio casi completamente ordenado, con la mayoría de los espines alineados. Esto es consistente con el hecho de que a temperaturas suficientemente por debajo de la temperatura crítica $T_c$, el sistema minimiza su energía adoptando un estado ferromagnético con magnetización espontánea distinta de cero. Las pequeñas fluctuaciones observadas (píxeles blancos aislados) corresponden a defectos térmicos poco probables.

- **A temperatura intermedia ($T = 2{,}0$):** se empiezan a formar dominios desordenados, aunque persiste una tendencia al orden. A esta temperatura, próxima pero aún inferior a $T_c$, las fluctuaciones térmicas comienzan a competir con la energía de interacción, lo cual induce la aparición de pequeñas regiones con espines opuestos. Sin embargo, todavía se conserva una magnetización neta no nula.

- **Cercano a la temperatura crítica ($T \approx 2{,}269$):** se evidencia un régimen crítico, caracterizado por la coexistencia de dominios de todos los tamaños y una alta fluctuación espacial. Esta configuración ilustra bien el comportamiento crítico del sistema: la correlación entre espines decae como una ley de potencias y no existe una longitud característica de correlación (la longitud de correlación diverge en el límite termodinámico).

- **A alta temperatura ($T = 4{,}0$):** el sistema se encuentra en un estado completamente desordenado, con espines distribuidos aleatoriamente. En esta fase paramagnética, las interacciones locales entre espines no logran competir contra el ruido térmico, lo cual conduce a una magnetización promedio nula y a la ausencia de estructura en la configuración.

Estos resultados son consistentes con el comportamiento teórico y las configuraciones muestran de manera cualitativa cómo la temperatura controla el orden en el sistema, pasando de una fase ferromagnética a una paramagnética a través de un punto crítico bien definido.

# 5. Conclusiones

A lo largo del trabajo hemos podido simular el modelo de Ising 2D y observar el estado del sistema para varias temperaturas, siendo esta visualización coherente con lo esperado. Simultáneamente, hemos podido ver la evolución de la magnetización, energía y calor específico frente a la temperatura para distintos tamaños de red, aproximándose también al comportamiento teórico, sobre todo en tamaños superiores de red. No obstante, el calor específico es la magnitud que más difiere con lo teórico, presentando picos por debajo de lo esperado, sugiriendo que esta magnitud puede ser más afectada por el tamaño finito de la

red o el número de pMC. Por último, se calculó la temperatura crítica y el exponente crítico, arrojando resultados válidos. Por todo ello, se han cumplimentado todos los objetivos propuestos.

# Referencias


[1] Hurtado, P. I. (2025). *Modelo de Ising: Una Introducción al Método Monte Carlo*. [Notas de curso].

[2] Hurtado, P. I. (2025). *Applications in Statistical Physics*. [Notas de curso].

[3] Onsager, L. (1944). *Crystal Statistics. I. A Two-Dimensional Model with an Order-Disorder Transition.* Physical Review.

[4] Yang, C. N. (1952). *The Spontaneous Magnetization of a Two-Dimensional Ising Model*. Physical Review.

[5] Landau, D. P., & Binder, K. (2009). *A Guide to Monte Carlo Simulations in Statistical Physics*. Cambridge University Press.