

Treball de Simulació 2 Física Estadística

Àlex Rodríguez Blanco (1635829)
Ferran Busquets González (1537584)

Curs 24-25

1 Magnetització i Energia

S'ha simulat una graella de 20×20 spinors que poden tenir un valor de -1 o 1. La configuració inicial és aleatòria. Seguint l'algoritme de Metropolis, s'ha simulat com canvia aquesta graella per a cada pas de temps, a una temperatura donada. A continuació es mostren els gràfics de la magnetització total del sistema, i l'energia mitjana per 3 temperatures diferents

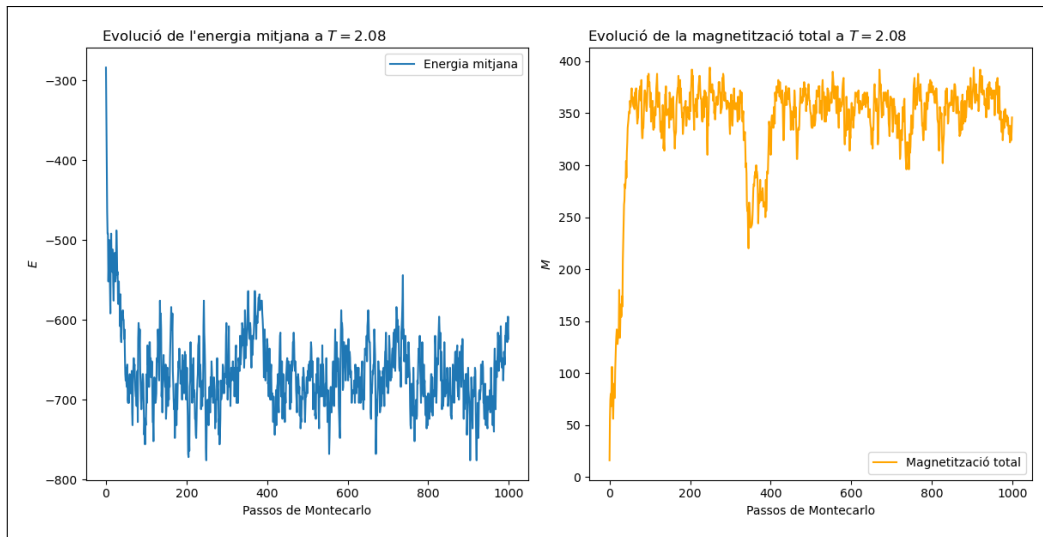


Figura 1: Gràfic de l'energia mitjana i la magnetització total del sistema en funció del temps (passes d l'algoritme), per a una temperatura de 2,08.

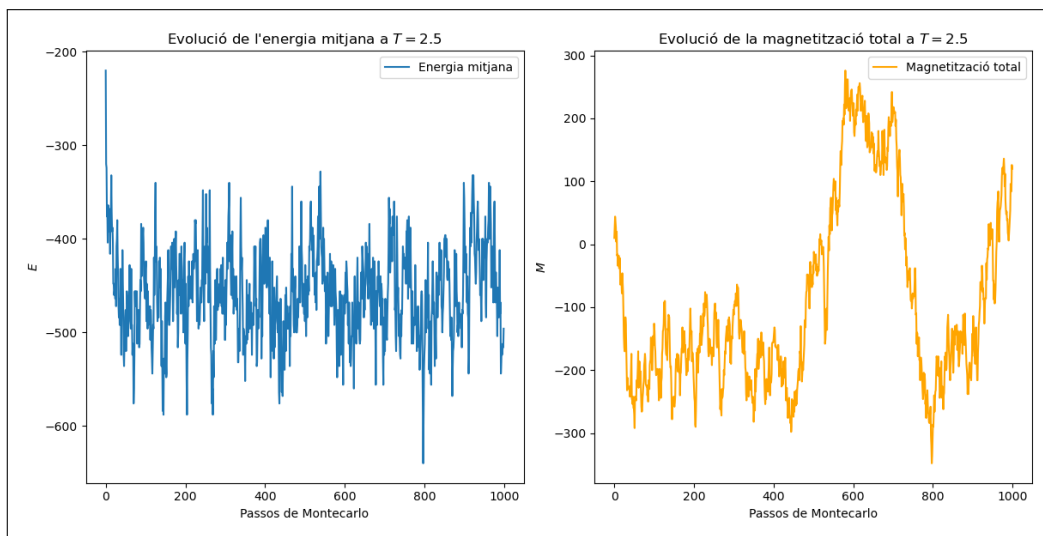


Figura 2: Gràfic de l'energia mitjana i la magnetització total del sistema en funció del temps (passes d l'algoritme), per a una temperatura de 2,50.

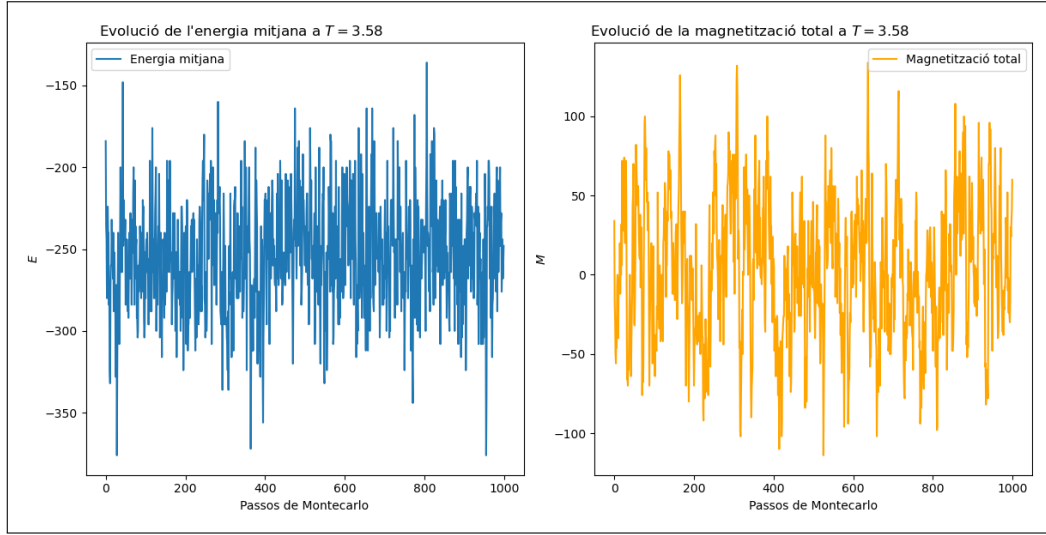


Figura 3: Gràfic de l'energia mitjana i la magnetització total del sistema en funció del temps (passes d l'algoritme), per a una temperatura de 3,58.

En aquests gràfics es pot veure que per a temperatures baixes l'energia del sistema es minimitza, i la magnetització tendeix a un valor constant. Com més gran sigui la temperatura, més fluctuaran aquests dos valors. Un cop passada la temperatura crítica, com es pot veure a la figura 3, la magnetització varia al voltant del 0. L'energia mitjana té el mateix comportament, però al voltant de ~ -200 , que correspon a l'energia que tindria si cada spinor tingués 2 veïns en cada direcció possible.

1.1 Dependència amb la Temperatura

Amb el codi de l'annex, s'ha trobat que la temperatura crítica del sistema és

$$T_c = 3,08$$

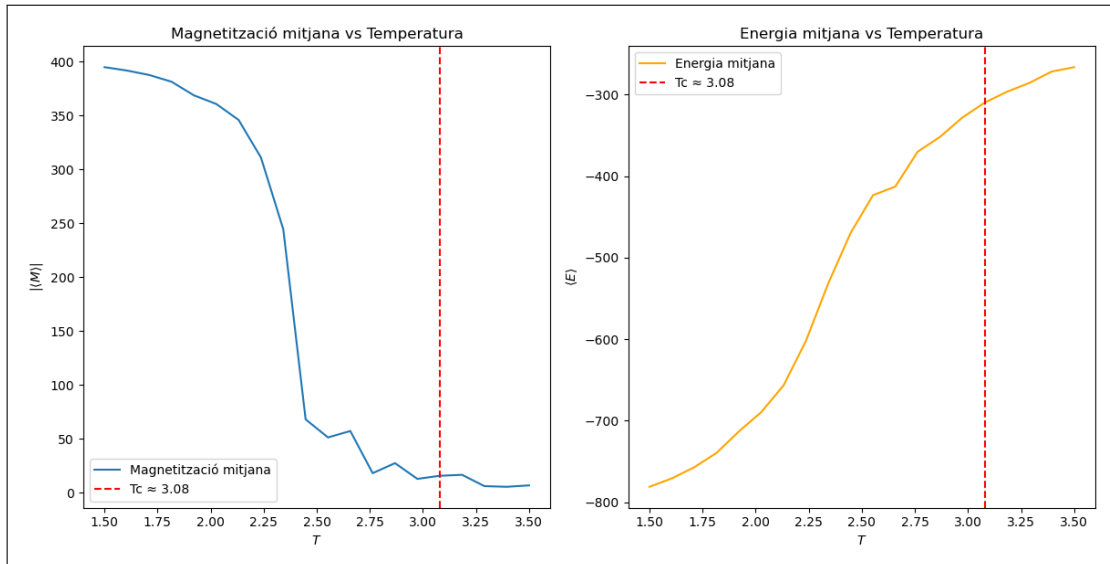


Figura 4: Gràfic de l'energia mitjana i la magnetització total del sistema en funció de la temperatura.

En la figura 4 es veu el comportament de la magnetització total, en valor absolut, i l'energia mitjana. Com s'esperava, per a temperatures més petites que la crítica la magnetització tendeix a 0, mentre que per temperatures més altes tendeix a un cert valor, que augmenta a mesura que disminueix la temperatura. Pel que fa a l'energia, es veu que passada la temperatura crítica tendeix a un valor proper a ~ -200 , que és el mateix que s'ha vist en la figura 3.

susceptibility = 23067.393762223997 specific heat= 5648.1877752576

2 Altres Paràmetres

2.1 Exponent crític β

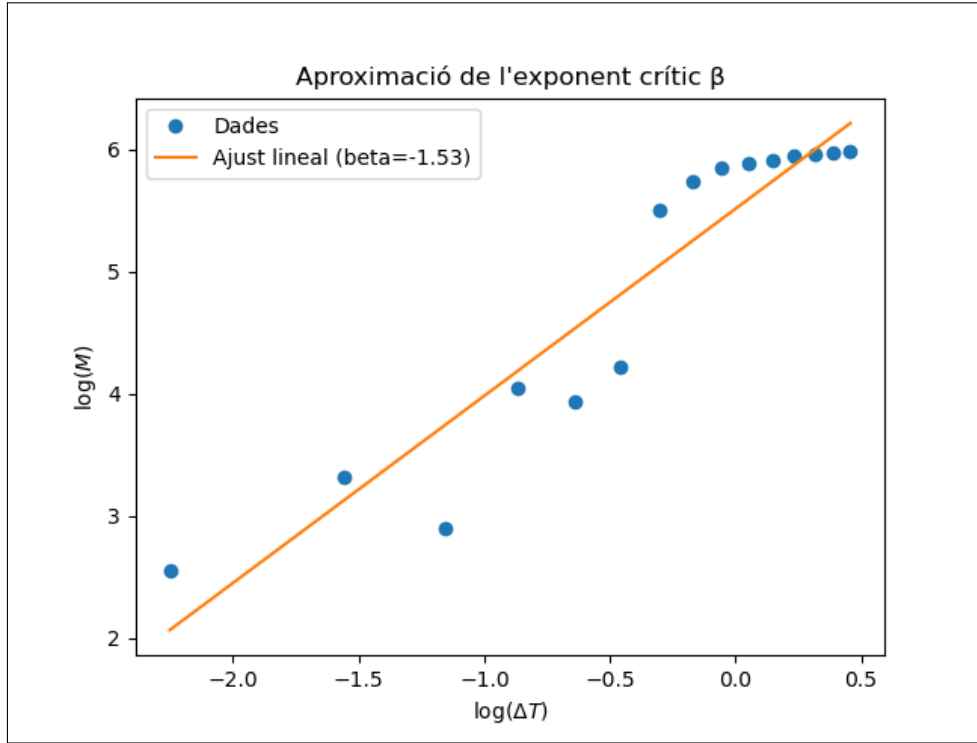


Figura 5: Gràfic de logaritme de la magnetització total en funció de la diferència de temperatures amb respecte a T_c .

Sabem per la teoria que la magnetització d'un sistema com el simulat hauria de tenir una expressió $M \propto |T - T_c|^\beta$. Aleshores, fent una regressió lineal del gràfic $\ln M$ vs $\ln \Delta T$ es pot obtenir aquest exponent amb el pendent de la recta. Amb el codi de l'annex s'ha trobat un resultat de

$$\beta = 1,53$$

Aquest valor hauria de ser de 0,5, que difereix molt de l'obtingut. Això no és sorprenent ja que aquesta simulació conté molt pocs spinors, i a més el model de *nearest neighbors* no és una aproximació prou bona.

2.2 Susceptibilitat magnètica

S'ha obtingut una susceptibilitat magnètica mitja, en l'interval de temperatures estudiat de

$$\chi = 23067,39$$

Per la teoria sabem que hauria de ser una asymptota, positiva per l'esquerra i negativa per la dreta, al voltant de la temperatura crítica.

2.3 Calor Específica

S'ha obtingut una susceptibilitat magnètica mitja, en l'interval de temperatures estudiat de

$$C = 5648,19$$

La calor específica, segons la tercera llei de la Termodinàmica, tendeix cap a 0, quan T tendeix cap a 0. Degut a l'aproximació que fem servir per simular el sistema, això no es compleix.

2.4 Estadística de Maxwell-Boltzmann

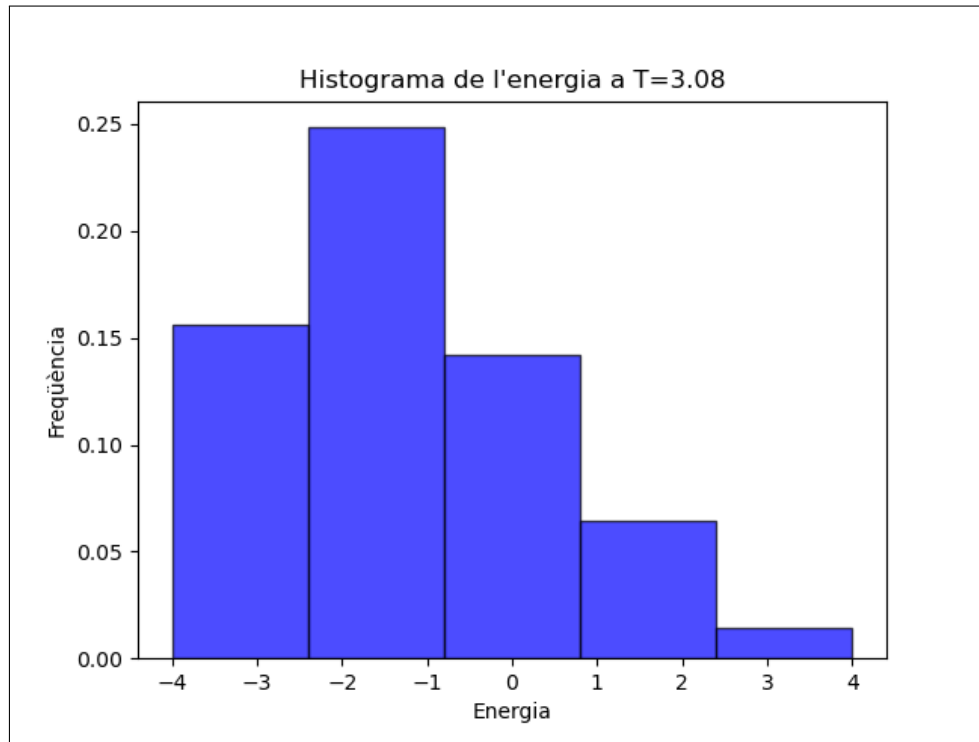


Figura 6: test

Aquest gràfic mostra un histograma de les energies de cada casell del sistema, a temperatura crítica. S'observa que segueix la mateixa forma que la distribució de Maxwell-Boltzmann. Això tampoc és cap sorpresa, ja que per la teoria sabem que sistemes macroscòpics en equilibri segueixen aquesta distribució d'energies per a cada partícula i com a probabilitat de l'energia total del sistema.

Codi

Aquest és el codi Python utilitzat per fer les figures i tots els càlculs.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Configuraci inicial
5 def inicialitza_matriu(L):
6     return np.random.choice([-1, 1], size=(L, L))
7
8 def energia(matriu, J=1):
9     energia_total = 0
10    L = matriu.shape[0]
11    for i in range(L):
12        for j in range(L):
13            # Spins ve ns amb condicions de contorn peri diques
14            ve ns = [
15                matriu[i, (j-1) % L],
16                matriu[i, (j+1) % L],
17                matriu[(i-1) % L, j],
18                matriu[(i+1) % L, j]
19            ]
20            energia_total -= J * matriu[i, j] * sum(ve ns)
21    return energia_total / 2 # Dividim entre 2 per no comptar interaccions
                               duplicades
22
23 def magnetitzacio(matriu):
24     return np.sum(matriu)
25
26 # Algorisme de Montecarlo amb criteri de Metropolis
27 def metropolis(matriu, T, J=1):
28     L = matriu.shape[0]
29     for _ in range(L * L): # Intentem actualitzar cada spin una vegada de mitjana
30         i, j = np.random.randint(0, L, size=2) # Seleccionem un spin aleatori
31         # Calculem el canvi d'energia si invertim aquest spin
32         ve ns = [
33             matriu[i, (j-1) % L],
34             matriu[i, (j+1) % L],
35             matriu[(i-1) % L, j],
36             matriu[(i+1) % L, j]
37         ]
38         delta_E = 2 * J * matriu[i, j] * sum(ve ns)
39         # Decidim si acceptem el canvi
40         if delta_E < 0 or np.random.rand() < np.exp(-delta_E / T):
41             matriu[i, j] *= -1 # Invertim el spin
42     return matriu
43
44 # Simulaci principal
45 def simula(L, N, T, J=1):
46     matriu = inicialitza_matriu(L)
47     energia_vals = []
48     magnetitzacio_vals = []
49
50     for _ in range(N):
51         matriu = metropolis(matriu, T, J)
52         energia_vals.append(energia(matriu, J))
53         magnetitzacio_vals.append(magnetitzacio(matriu))
54
55     return matriu, energia_vals, magnetitzacio_vals
56
57 # Estudi en funci de la temperatura
58 def estudi_temperatures(L, N, temperatures, J=1):
59     magnetitzacions = []
60     energies = []
61
62     for T in temperatures:
```

```

63     _, energia_vals, magnetitzacio_vals = simula(L, N, T, J)
64     magnetitzacions.append(np.mean(magnetitzacio_vals[int(N/2):])) # Equilibri
65     energies.append(np.mean(energia_vals[int(N/2):])) # Equilibri
66
67     return magnetitzacions, energies
68
69 # Determinaci de la temperatura cr tica
70 def determina_tc(temperatures, magnetitzacions):
71     gradients = np.gradient(np.abs(magnetitzacions), np.abs(temperatures))
72     return temperatures[np.argmax(gradients)]
73
74 # Aproximaci de l'exponent cr tic beta
75 def calcula_exponent_critic(temperatures, magnetitzacions, Tc):
76     delta_T = Tc - np.array(temperatures)
77     delta_T = delta_T[delta_T > 0] # Considerem nom s temperatures menors que Tc
78     mags = np.array(np.abs(magnetitzacions))[:len(delta_T)]
79
80     # Ajust lineal log-log per beta
81     log_delta_T = np.log(delta_T)
82     log_mags = np.log(mags)
83     coeficients = np.polyfit(log_delta_T, log_mags, 1)
84     beta = coeficients[0]
85
86     plt.figure()
87     plt.plot(log_delta_T, log_mags, 'o', label='Dades')
88     plt.plot(log_delta_T, np.polyval(coeficients, log_delta_T), label=f'Ajust lineal (beta={beta:.2f})')
89     plt.xlabel(r'log($ T $)')
90     plt.ylabel(r'log($M $)')
91     plt.title("Aproximaci de l'exponent cr tic ")
92     plt.legend()
93     plt.show()
94
95     return beta
96
97 # Evoluci de l'energia mitjana i la magnetitzaci total en funci de N
98 def evolucio_en_N(L, N, T, J=1):
99     _, energia_vals, magnetitzacio_vals = simula(L, N, T, J)
100
101     plt.figure(figsize=(12, 6))
102     plt.subplot(1, 2, 1)
103     plt.plot(range(N), energia_vals, label='Energia mitjana')
104     plt.title(r"Evoluci de l'energia mitjana a $T=%s$" % round(T, 2))
105     plt.xlabel('Passos de Montecarlo')
106     plt.ylabel(r'$E$')
107     plt.legend()
108
109     plt.subplot(1, 2, 2)
110     plt.plot(range(N), magnetitzacio_vals, label='Magnetitzaci total', color='orange')
111     plt.title(r'Evoluci de la magnetitzaci total a $T= %s$' % round(T, 2))
112     plt.xlabel('Passos de Montecarlo')
113     plt.ylabel(r'$M$')
114     plt.legend()
115
116     plt.tight_layout()
117     plt.show()
118
119 # Objectius opcionals avan ats
120 def calcula_susceptibilitat(magnetitzacions, temperatures):
121     return np.var(magnetitzacions) / np.mean(temperatures)
122
123 def calcula_calor_especifica(energies, temperatures):
124     return np.var(energies) / (np.mean(temperatures)**2)
125
126 def histograma_energia(L, N, T, J=1):

```

```

127     matriu, _, _ = simula(L, N, T, J)
128     L = matriu.shape[0]
129     energies = []
130     for i in range(L):
131         for j in range(L):
132             ve ns = [
133                 matriu[i, (j-1) % L],
134                 matriu[i, (j+1) % L],
135                 matriu[(i-1) % L, j],
136                 matriu[(i+1) % L, j]
137             ]
138             energies.append(-J * matriu[i, j] * sum(ve ns))
139
140     plt.hist(energies, bins=5, density=True, alpha=0.7, color='blue', edgecolor='
black')
141     plt.title('Histograma de l\'energia a T={:.2f}'.format(T))
142     plt.xlabel('Energia')
143     plt.ylabel('Freq  ncia')
144     plt.show()
145
146 # Configuraci  de la simulaci
147 L = 20 # Dimensions de la matriu
148 N = 1000 # Nombre de passos de simulaci
149 temperatures = np.linspace(1.5, 3.5, 20) # Diferents temperatures per a l'estudi
150
151 # Simulaci en funci  de les temperatures
152 magnetitzacions, energies = estudi_temperatures(L, N, temperatures)
153 Tc = determina_tc(temperatures, magnetitzacions)
154
155 # Resultats
156 print(f"Temperatura cr tica aproximada: Tc = {Tc}")
157 print(f"Susceptibilitat: {calcula_susceptibilitat(magnetitzacions, temperatures)}")
158 print(f"Calor espec fica: {calcula_calor_especifica(energies, temperatures)}")
159
160 # Gr fics i histograma
161 plt.figure(figsize=(12, 6))
162 plt.subplot(1, 2, 1)
163 plt.plot(temperatures, np.abs(magnetitzacions), label='Magnetitzaci mitjana')
164 plt.axvline(Tc, color='red', linestyle='--', label=f'Tc {Tc:.2f}')
165 plt.title('Magnetitzaci mitjana vs Temperatura')
166 plt.xlabel(r'$T$')
167 plt.ylabel(r'$\angle M \angle$')
168 plt.legend()
169
170 plt.subplot(1, 2, 2)
171 plt.plot(temperatures, energies, label='Energia mitjana', color='orange')
172 plt.axvline(Tc, color='red', linestyle='--', label=f'Tc {Tc:.2f}')
173 plt.title('Energia mitjana vs Temperatura')
174 plt.xlabel(r'$T$')
175 plt.ylabel(r'$\angle E \angle$')
176 plt.legend()
177
178 plt.tight_layout()
179 plt.show()
180
181 # Histograma a temperatura espec fica
182 histograma_energia(L, N, T=Tc, J=1)
183
184 # Exponent cr tic beta
185 beta = calcula_exponent_critic(temperatures, magnetitzacions, Tc)
186 print(f"Exponent cr tic beta: {beta:.2f}")
187
188 # Evoluci  en N
189 print("Evoluci  de l'energia i la magnetitzaci  per a diverses temperatures")
190 Temp_plot= [Tc-1, 2.5, Tc+0.5]
191 for t in range(len(Temp_plot)):

```

```
192     evolucion_en_N(L, N, T= Temp_plot[t], J= 1)
```