Computational applied mathematics

by

ROME KEIZER MEDINA DELMO

29th May 2024

Table of contents

Question 3 *Simulations of the Ising model (Unit 10IM)*

Part (a)

(i) Let the energy of state 1 and state 2 be $E_1 = 0$ and $E_2 = \varepsilon$, respectively. From Subsection 3.2.3 of the Unit 10IM, equation (9) shows that the mean energy $\langle E \rangle$

$$\langle E \rangle = \sum_{\alpha=1}^{2^N} E_\alpha P(E_\alpha),$$

where $P(E_\alpha)$ is the probability that the system in a state α has the energy E_α. Since the two-state system is in thermodynamic equilibrium according to Boltzmann's distribution, from Subsection 3.2.2, the probability $P(E_\alpha)$ is described by equation (8) that is

$$P(E_\alpha) = \frac{e^{-E_\alpha/(k_B T)}}{Z},$$

where the partition function Z is

$$Z = \sum_{\alpha=1}^{2^N} e^{-E_\alpha/(k_B T)}.$$

The system has two states so it must be that $N = 1$. Given that the partition function is

$$Z = \sum_{\alpha=1}^{2^N} e^{-E_\alpha/(k_B T)} = e^{-E_1/(k_B T)} + e^{-E_2/(k_B T)} = e^{-0/(k_B T)} + e^{-\varepsilon/(k_B T)} = 1 + e^{-\varepsilon/(k_B T)},$$

then the mean energy of the two-state system is

$$\langle E \rangle = \sum_{\alpha=1}^{2} E_\alpha P(E_\alpha)$$

$$= E_1 P(E_1) + E_2 P(E_2)$$

$$= (0) P(E_1) + \varepsilon\, P(E_2)$$

$$= \varepsilon\, P(E_2)$$

$$= \frac{\varepsilon\, e^{-\varepsilon/(k_B T)}}{1 + e^{-\varepsilon/(k_B T)}}.$$

(ii) In Subsection 3.2.3, equation (14) shows that the mean-square energy $\langle E^2 \rangle$ is

$$\langle E^2 \rangle = \sum_{\alpha=1}^{2^N} E_\alpha^2 \, P(E_\alpha).$$

Using the information from part (i), the mean-square energy of the two-state system is

$$\langle E^2 \rangle = \sum_{\alpha=1}^{2^N} E_\alpha^2 \, P(E_\alpha)$$

$$= E_1^2 P(E_1) + E_2^2 P(E_2)$$

$$= (0)^2 P(E_1) + \varepsilon^2 \, P(E_2)$$

$$= \varepsilon^2 \, P(E_2)$$

$$= \frac{\varepsilon^2 \, e^{-\varepsilon/(k_B T)}}{1 + e^{-\varepsilon/(k_B T)}}.$$

(iii) Equation (14) also states that the specific heat capacity is defined by

$$c = \frac{1}{k_B T^2 N} (\langle E^2 \rangle - \langle E \rangle^2).$$

Given that the number of atoms is $N = 1$, the specific heat capacity can be written as

$$c = \frac{1}{k_B T^2 N} (\langle E^2 \rangle - \langle E \rangle^2)$$

$$= \frac{1}{k_B T^2} \left(\varepsilon^2 \, P(E_2) - \varepsilon^2 \left(P(E_2) \right)^2 \right)$$

$$= \frac{1}{k_B T^2} \varepsilon^2 \, P(E_2) \left(1 - P(E_2) \right)$$

$$= \frac{1}{k_B T^2} \varepsilon^2 \, P(E_2) \left(\frac{1 + e^{-\varepsilon/(k_B T)}}{1 + e^{-\varepsilon/(k_B T)}} - \frac{e^{-\varepsilon/(k_B T)}}{1 + e^{-\varepsilon/(k_B T)}} \right)$$

$$= \frac{1}{k_B T^2} \frac{\varepsilon^2 \, e^{-\varepsilon/(k_B T)}}{1 + e^{-\varepsilon/(k_B T)}} \left(\frac{1}{1 + e^{-\varepsilon/(k_B T)}} \right)$$

$$= \frac{1}{k_B T^2} \frac{\varepsilon^2 \, e^{-\varepsilon/(k_B T)}}{(1 + e^{-\varepsilon/(k_B T)})^2} \, ,$$

as required.

Part (b)

The code for parts (i) and (ii) were combined, as shown in Figure 1a. This was done because the simulated and analytical plots are plotted together, as seen in Figure 1b. It was decided not to include a separate graph showing only the simulated plots without the analytical one. This decision ensures consistency because if the graph was regenerated to include the analytical plot, the simulated outcome for part (ii) would change. This alteration would invalidate the comparison intended for part (iii). Therefore, Figures 1a and 1b serve as complete representations, addressing both parts (i) and (ii).

```
%reset -f
import numpy as np
import matplotlib.pyplot as plt
from numba import njit

@njit
def metropolis_sweep_ising_2D(spins, T):
    L = spins.shape[0]
    for i in np.random.permutation(L):
        for j in np.random.permutation(L):
            DeltaE = 2 * spins[i, j] * (spins[i - 1, j] + spins[(i + 1) % L, j] +
                                        spins[i, j - 1] + spins[i, (j + 1) % L])
            if np.exp(-DeltaE / T) > np.random.random():
                spins[i, j] = -spins[i, j]

def magnetic_moment_ising_2D(spins):
    S = 0.0
    for i in range(spins.shape[0]):
        for j in range(spins.shape[0]):
            S += spins[i, j]
    return S

# Required parameters
relax_sweeps = 500
sweeps = 2000
temperatures = np.arange(0.2, 4.0, 0.1)
plt.figure(figsize=(12, 6))
L_values = [30, 40, 50, 60]

# Plotting of m vs T for different lattice sizes
for L in L_values:
    N = L * L
    magnetisation_T = []
    for T in temperatures:
        spins = np.ones((L, L))
        magnetic_moment_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_2D(spins, T)
            magnetic_moment_sweep.append(magnetic_moment_ising_2D(spins) / N)
        magnetisation_T.append(np.mean(magnetic_moment_sweep[relax_sweeps:]))
    plt.plot(temperatures, magnetisation_T, ":o", label=f'N = {L} X {L}')

```

```python
# Calculation of critical temperature, T_c
J = 1
k_B = 1
T_c = (2*J) / (k_B*np.log(1+np.sqrt(2)))

# Mathematically exact magnetisation per spin
def m(T):
    if T <= T_c:
        return (1 - (np.sinh( (2*J)/(k_B*T)) )**(-4))**(1/8)
    else:
        return 0

analytical_m = []

for T in temperatures:
    # Calculate magnetisation for the current temperature
    m_values = m(T)
    # Append the result to the list
    analytical_m.append(m_values)

# Plotting the graph
plt.plot(temperatures, analytical_m, '-o', color='black', label='Analytical')
plt.xlabel('Temperature')
plt.ylabel('Magnetisation per spin')
plt.title('Magnetisation per Spin vs Temperature for Different Lattice Sizes')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 1a Code used to simulate the 2D Ising Model using the Metropolis-Hastings algorithm to produce four plots of magnetisation per spin as a function of temperature for different lattice sizes and a plot of the mathematically exact magnetisation per spin

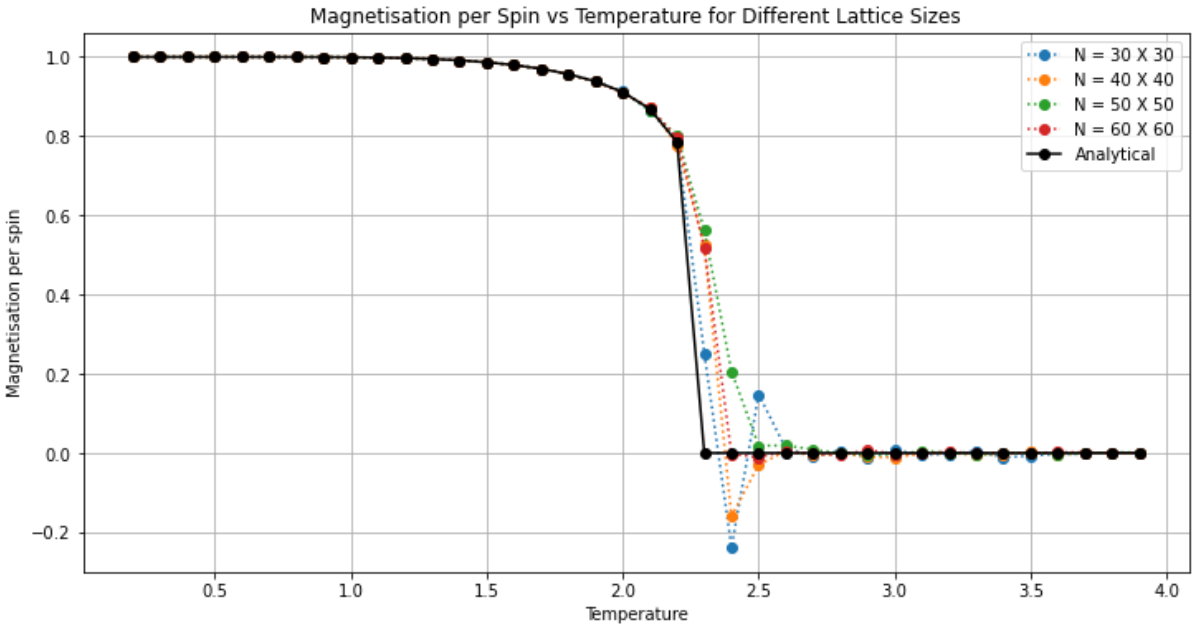


Figure 1b A single graph presenting four simulation plots of magnetisation per spin as a function of temperature from $T = 0.2$ to $T = 4.0$ (coloured dashed lines) and a plot of the mathematically exact magnetisation per spin (solid black line) described by the equation in the question

(i) From Figure 1a, Lines 7 to 21 show the required user-defined functions which were imported from Computer session 10IM.1. The `metropolis_sweep_ising_2D` function performs a Monte Carlo sweep on a 2D system of N spins when the external magnetic field is $B = 0$. The `magnetic_moment_ising_2D` calculates the magnetisation of the system. After defining the required parameters required by the question in Lines 23 to 28, the two imported functions are used to produce plots of magnetisation per spin against temperature, between $T = 0.2$ to $T = 4.0$ for lattices sizes 30×30, 40×40, 50×50, and 60×60.

Figure 1b shows four simulation plots of magnetisation per spin as a function of temperature in a single graph. The lattice sizes 30×30, 40×40, 50×50, and 60×60 are represented by the dashed blue, orange, green, and red lines respectively.

(ii) Lines 43 to 46 of Figure 1a defined the value of the critical temperature T_c. This was used in Lines 48 to 61 where the mathematically exact magnetisation per spin is calculated to be plotted against the temperature range defined in part (i).

Figure 1b presents the exact values, called the analytical result, plotted on top of the simulated results. The analytical plot is represented by the solid black line.

(iii) At low temperatures, the magnetisation per spin of the simulated results is approximately constant. As the temperature increases, between $T = 2.0$ and $T = 2.5$, magnetisation per spin of both simulated and analytical results sharply declines, which indicates phase transition. The simulation plots decline less steeply than the analytical plots. This means that the critical temperatures of the simulated results are slightly greater than but close to $T_c = 2J/(k_B \ln(1 + \sqrt{2})) \simeq 2.26918$. After about $T = 2.6$, the simulated results start to conform again with the analytical result where magnetisation per spin remains approximately zero.

Part (c)

(i) From Subsection 3.1 of Unit 10IM, equation (6) presents the total energy for the one-dimensional Ising model with N spins $\mathbf{s} = (s_1, s_2, \dots, s_N)$ where J is the exchange constant and B is the external magnetic field component pointing at direction **d**. Equation 6 can be written as

$$E = -J\sum_{i=1}^{N} s_i s_{i+1} - B\sum_{i=1}^{N} s_i \,,$$

where $s_{N+1} = s_1$ because of the imposed periodic boundary conditions. Given that the state of the system is $\mathbf{s}_a$when the spin is s_k, then the energy in this state can be written as

$$E_a = -J(s_1 s_2 + s_2 s_3 + \cdots + s_{k-1} s_k + s_k s_{k+1} + \cdots + s_N s_1)$$

$$-B(s_1 + s_2 + \cdots + s_{k-1} + s_k + \cdots + s_N).$$

Assigning $\mathbf{s}_b$ as the state of the system corresponding to the flipped spin where $s_k \to -s_k$, then the energy in this state is

$$E_b = -J(s_1 s_2 + s_2 s_3 + \cdots - s_{k-1} s_k - s_k s_{k+1} + \cdots + s_N s_1)$$

$$-B(s_1 + s_2 + \cdots + s_{k-1} - s_k + \cdots + s_N).$$

The change in energy when the spin s_k is flipped is

$$E_b - E_a = 2J\, s_{k-1} s_k + 2J\, s_k s_{k+1} + 2B\, s_k$$

$$= 2J s_k\, (s_{k-1} + s_{k+1}) + 2B\, s_k$$

$$= 2s_k(\, J\, (s_{k-1} + s_{k+1}) + B),$$

the terms s_{k-1} and s_{k+1} are the nearest neighbours (n. n) to spin s_k, so $(s_{k-1} + s_{k+1})$ can be alternatively expressed as

$$(s_{k-1} + s_{k+1}) = \sum_{i \text{ n.n.of } k} s_i \,.$$

Therefore, the change in energy is

$$E_b - E_a = 2s_k\left(J \sum_{i \text{ n.n.of } k} s_i \; + B\right)$$

as required.

(ii) The code used to produce four plots of the magnetisation per spin against temperature when the external magnetic field is $B = 0.1$ is given in Figure 2a. This is similar to the code used in part (b) but includes a few changes. This includes the user-defined function in Lines 8 to 14, where the Monte Carlo sweep for one-dimension (1D), imported from Computer session 10IM.1, had been changed to include B. Initially, `DeltaE` in Line 11 follows the equation

$$E_b - E_a = 2s_k\, J\, (s_{k-1} + s_{k+1}),$$

and now this was updated to follow the equation ascertained from part (c)(i), that is

$$\begin{aligned} E_b - E_a &= 2s_k(\, J\, (s_{k-1} + s_{k+1}) + B) \\ &= 2s_k\, J\, (s_{k-1} + s_{k+1}) + 2s_k B. \end{aligned}$$

Therefore, Line 11 was updated by adding the term $2s_k B$. This updated function was renamed as `metropolis_sweep_ising_1D_nonzero_B`, and the argument included $B = 0.1$.

Moreover, the user-defined function `magnetic_moment_ising_2D` from part (b) imported from Computer session 10IM.1 was altered for the 1D case. This was done by removing the inner loop "`for j in range(spins.shape[0]):`" and summing the spins along a single dimension using so that "`S += spins[i]`". Therefore, the resulting code in Lines 18 to 22 of Figure 2a computes the total magnetic moment in a 1D lattice.

After defining the required parameters in Lines 25 to 29, the two functions were used to produce plots of magnetisation per spin against temperature, between $T = 0.2$ to $T = 4.0$ for lattice sizes $N = 8, 16, 32, 64$.

The mathematically exact magnetisation per spin is defined and calculated in Lines 45 to 57. This was then plotted against the required temperature range, on the same graph as the four simulated plots.

```
%reset -f
import numpy as np
import matplotlib.pyplot as plt
from numba import njit

@njit
# Performs a Monte Carlo sweep on a 1D system of N spins, with non-zero B
def metropolis_sweep_ising_1D_nonzero_B(spins,T ,B = 0.1):
    N = len(spins)
    for i in np.random.permutation(N):  # Spins are selected in random order
        DeltaE = 2 * spins[i] * (spins[i - 1] + spins[(i + 1) % N]) + (2 * spins[i] * B)

        if np.exp(-DeltaE / T) > np.random.random():  # Metropolis-Hastings step
            spins[i] = -spins[i]

@njit
# Updated code from 2D to 1D
def magnetic_moment_ising_1D(spins):
    S = 0.0
    for i in range(spins.shape[0]):
        S += spins[i]
    return S

# Required parameters
relax_sweeps = 500
sweeps = 2000
temperatures = np.arange(0.2, 4.0, 0.1)
plt.figure(figsize=(12, 6))
L_values = [8, 16, 32, 64]

# Plotting of m vs T for different lattice sizes
for L in L_values:
    N = L
    magnetisation_T = []
    for T in temperatures:
        spins = np.ones((L))
        magnetic_moment_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_1D_nonzero_B(spins, T)
            magnetic_moment_sweep.append(magnetic_moment_ising_1D(spins) / N)
        magnetisation_T.append(np.mean(magnetic_moment_sweep[relax_sweeps:]))
    plt.plot(temperatures, magnetisation_T, ":o", label=f'N = {L}')
```

```
43
44 # Mathematically exact magnetisation per spin, where B = 0.1
45 J = 1
46 k_B = 1
47 def m(T, B = 0.1):
48     beta = 1 / (k_B * T)
49     return (np.sinh(beta*B)) / (((np.cosh(beta*B))**2) - 2 * np.exp(-2*beta*J) * np.sinh(2*beta*J))**(1/2)
50
51 analytical_m = []
52
53 for T in temperatures:
54     # Calculate magnetisation for the current temperature
55     m_values = m(T)
56     # Append the result to the list
57     analytical_m.append(m_values)
58
59 # Plotting the graph
60 plt.plot(temperatures, analytical_m, '-o', color='black', label='Analytical')
61 plt.xlabel('Temperature')
62 plt.ylabel('Magnetisation per spin')
63 plt.title('Magnetisation per Spin vs Temperature for Different Lattice Sizes, B= 0.1')
64 plt.legend()
65 plt.grid(True)
66 plt.show();
```

Figure 2a Code used to simulate the 1D Ising Model using the Metropolis-Hastings algorithm to produce four plots of magnetisation per spin as a function of temperature for different lattice sizes and a plot of the mathematically exact magnetisation per spin, when $B = 0.1$

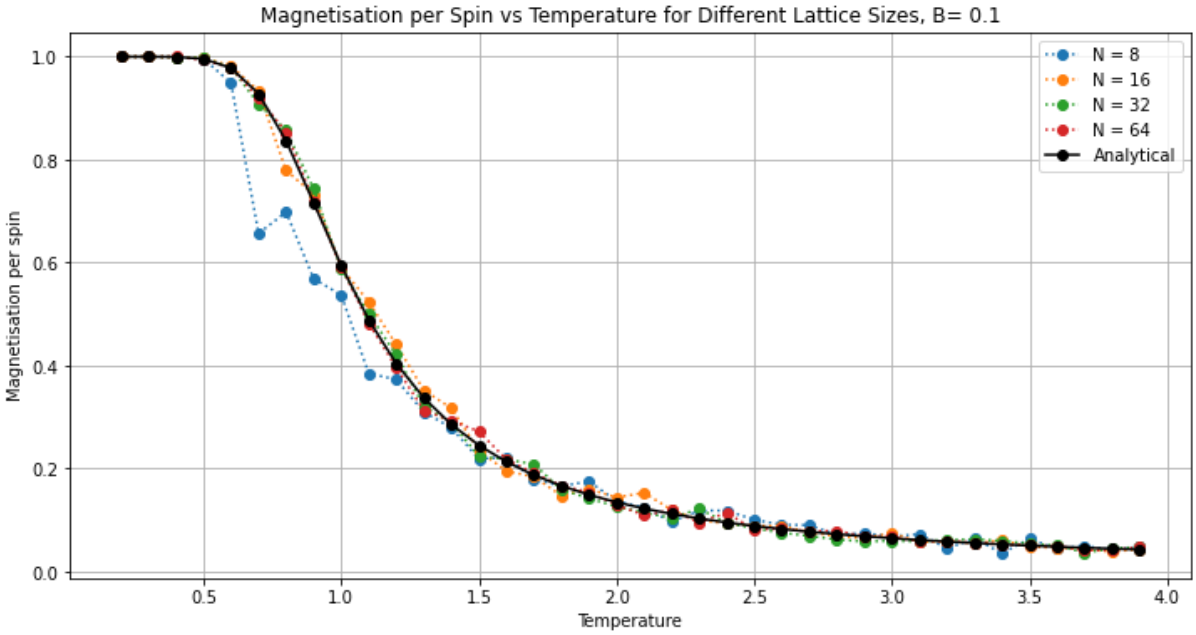


Figure 2b A single graph presenting four simulation plots of magnetisation per spin as a function of temperature from $T = 0.2$ to $T = 4.0$ (coloured dashed lines) and a plot of the mathematically exact magnetisation per spin (solid black line) described by the equation in the question, when $B = 0.1$

Figure 2b presents four simulation plots of magnetisation per spin as a function of temperature in a single graph. The lattice sizes $N = 8, 16, 32, 64$ are represented by the dashed blue, orange, green, and red lines respectively. The mathematically exact magnetisation per spin as a function of temperature was plotted on top of the simulated results. This analytical plot is represented by the solid black curve.

The code used to produce four plots of the magnetisation per spin against temperature when the external magnetic field is $B = 0.5$ is given in Figure 2c. This is similar to Figure 2a but the arguments including B were updated from holding the value 0.1 to 0.5; these changes were made in Lines 7 and 45 of Figure 2c. This nearly identical code is repeated here for completion.

```
1  %reset -f
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from numba import njit
5
6  @njit
7  def metropolis_sweep_ising_1D_nonzero_B(spins,T ,B = 0.5):
8      N = len(spins)
9      for i in np.random.permutation(N):  # Spins are selected in random order
10         DeltaE = 2 * spins[i] * (spins[i - 1] + spins[(i + 1) % N]) + (2*spins[i]*B)
11
12         if np.exp(-DeltaE / T) > np.random.random():  # Metropolis-Hastings step
13             spins[i] = -spins[i]
14
15 @njit
16 def magnetic_moment_ising_1D(spins):
17     S = 0.0
18     for i in range(spins.shape[0]):
19         S += spins[i]
20     return S
21
22 # Required parameters
23 relax_sweeps = 500
24 sweeps = 2000
25 temperatures = np.arange(0.2, 4.0, 0.1)
26 plt.figure(figsize=(12, 6))
27 L_values = [8, 16, 32, 64]
28
29 # Plotting of m vs T for different lattice sizes
30 for L in L_values:
31     N = L
32     magnetisation_T = []
33     for T in temperatures:
34         spins = np.ones((L))
35         magnetic_moment_sweep = []
36         for sweep in range(sweeps + relax_sweeps):
37             metropolis_sweep_ising_1D_nonzero_B(spins, T)
38             magnetic_moment_sweep.append(magnetic_moment_ising_1D(spins) / N)
39         magnetisation_T.append(np.mean(magnetic_moment_sweep[relax_sweeps:]))
40     plt.plot(temperatures, magnetisation_T, ":o", label=f'N = {L}')
41
```

```python
# Mathematically exact magnetisation per spin
J = 1
k_B = 1
def m(T, B = 0.5):
    beta = 1 / (k_B * T)
    return (np.sinh(beta*B)) / (((np.cosh(beta*B))**2) - 2 * np.exp(-2*beta*J) * np.sinh(2*beta*J))**(1/2)

analytical_m = []

for T in temperatures:
    # Calculate magnetisation for the current temperature
    m_values = m(T)
    # Append the result to the list
    analytical_m.append(m_values)

# Plotting the graph
plt.plot(temperatures, analytical_m, '-o', color='black', label='Analytical')
plt.xlabel('Temperature')
plt.ylabel('Magnetisation per spin')
plt.title('Magnetisation per Spin vs Temperature for Different Lattice Sizes, B= 0.5')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 2c Code used to simulate the 1D Ising Model using the Metropolis-Hastings algorithm to produce four plots of magnetisation per spin as a function of temperature for different lattice sizes and a plot of the mathematically exact magnetisation per spin, when $B = 0.5$

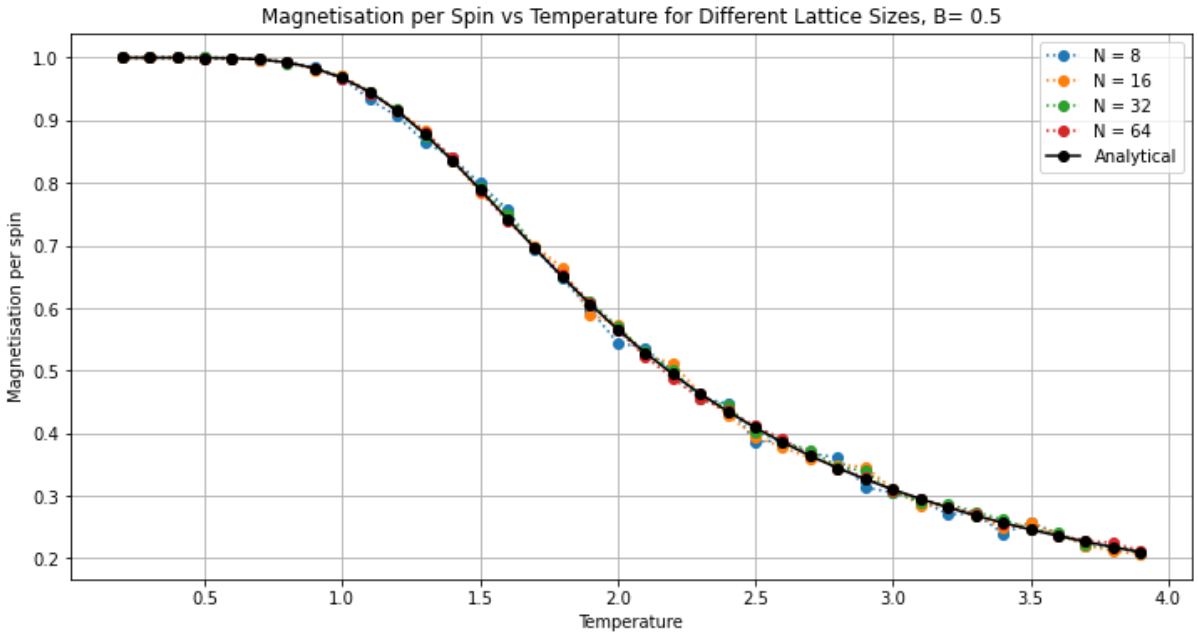


Figure 2d A single graph presenting four simulation plots of magnetisation per spin as a function of temperature from $T = 0.2$ to $T = 4.0$ (coloured dashed lines) and a plot of the mathematically exact magnetisation per spin (solid black line) described by the equation in the question, when $B = 0.5$

In Figure 2d, in the case where $B = 0.5$, four simulation plots of magnetisation per spin as a function of temperature in a single graph are presented. The lattice sizes $N = 8, 16, 32, 64$ are represented by the dashed blue, orange, green, and red lines respectively. The mathematically exact magnetisation per spin as a function of temperature was plotted on top of the simulated results, shown by the solid black line.

Part (d)

(i) From Section 4 of Unit 10IM, equation (15) presents the total energy for the two-dimensional Ising model, that is

$$E = -J\sum_{\langle i,j\rangle} s_i s_j - B\sum_i s_i \, .$$

Since $\langle i,j\rangle$ means the sum over all the nearest neighbours, then the energy state of a system $\mathbf{s}_a$ with a spin s_k at $s_{k,k}$ can be written as

$$E_a = -J\big(s_{1,1}s_{1,2} + s_{1,1}s_{2,1} + \cdots + s_{k,k}s_{(k+i),k} + s_{k,k}s_{k,(k+j)} + s_{k,k}s_{(k-i),k} + s_{k,k}s_{k,(k-j)}$$

$$+ \cdots + s_{1,1}s_{N,N}\big) - B\big(s_{1,1} + \cdots + s_{k,k} + \cdots + s_{N,N}\big).$$

Let $\mathbf{s}_b$ be state of the system corresponding to the flipped spin where $s_k \to -s_k$ at $s_{k,k}$ so that $s_{k,k} \to -s_{k,k}$. The energy in this state is

$$E_b = -J\big(s_{1,1}s_{1,2} + s_{1,1}s_{2,1} + \cdots - s_{k,k}s_{(k+i),k} - s_{k,k}s_{k,(k+j)} - s_{k,k}s_{(k-i),k} - s_{k,k}s_{k,(k-j)}$$

$$+ \cdots + s_{1,1}s_{N,N}\big) - B\big(s_{1,1} + \cdots - s_{k,k} + \cdots + s_{N,N}\big).$$

Therefore, the change in energy when the spin s_k is flipped is

$$E_b - E_a = 2J\big(s_{k,k}s_{(k+i),k} + s_{k,k}s_{k,(k+j)} + s_{k,k}s_{(k-i),k} + s_{k,k}s_{k,(k-j)}\big) + 2B\big(s_{k,k}\big)$$

$$= 2s_{k,k}\big(J\left(s_{(k+i),k} + s_{k,(k+j)} + s_{(k-i),k} + s_{k,(k-j)}\right) + B\big).$$

Since the terms $s_{(k+i),k}$, $s_{k,(k+j)}$, $s_{(k-i),k}$ and $s_{k,(k-j)}$ are the nearest neighbours (n.n) to spin s_k, so it can be alternatively expressed as

$$\big(s_{(k+i),k} + s_{k,(k+j)} + s_{(k-i),k} + s_{k,(k-j)}\big) = \sum_{i \text{ n.n.of } k} s_i \, .$$

Hence, valid in an arbitrary number of dimensions, the change in energy is

$$E_b - E_a = 2s_k\left(J\sum_{i \text{ n.n.of } k} s_i + B\right).$$

(ii) The code used to create the single graph containing four plots of the meant total energy as a function of temperature is presented in Figure 3a. In Figure 3a, Lines 7 to 15 show the modified function from Computer session 10IM.1 that performs a Monte Carlo sweep on a 2D system of N spins; an extra term was added to `DeltaE` in Line 13 which corresponds to the additional term $2\ s_k\ B$ from the change in energy derived in part (d) (i). This was done to include the effect of an external magnetic field B.

Similarly, Lines 18 to 25 show the modified function from Computer session 10IM.1 that calculates the energy of the 2D Ising model. Following equation (16) of Unit 10IM, an additional term was appended to the calculation of energy, shown in Line 24, that arises now that B is non-zero.

The required parameters were defined in Lines 28 to 35 and the process of calculating and plotting the mean total energy against the temperature was demonstrated in Lines 38 to 57. The mean total energy was plotted against a temperature range of $T = 0.1$ to $T = 10$ for varying external magnetic field $B = 0.0, 0.2, 0.4, 0.6$. This was demonstrated for a constant lattice size of $N = 32 \times 32$. The output of the code is shown in Figure 3b.

```
1  %reset -f
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from numba import njit
5
6  @njit
7  def metropolis_sweep_ising_2D_nonzero_B(spins, T, B):
8      L = spins.shape[0]
9      for i in np.random.permutation(L):
10         for j in np.random.permutation(L):
11             DeltaE = 2 * spins[i, j] * (spins[i - 1, j] + spins[(i + 1) % L, j] +
12                                         spins[i, j - 1] + spins[i, (j + 1) % L])\
13             + 2 * spins[i, j] * B
14             if np.exp(-DeltaE / T) > np.random.random():
15                 spins[i, j] = -spins[i, j]
16
17 @njit
18 def energy_ising_2D_nonzero_B(spins, B):
19     L = spins.shape[0]
20     E = 0.0
21     for i in range(L):
22         for j in range(L):
23             E += -spins[i, j] * (spins[i - 1, j] + spins[i, j - 1])\
24             - B * (spins[i, j])
25     return E
26
```

```
# Required parameters
L = 32
relax_sweeps = 500
sweeps = 2000
temperatures = np.arange(0.1, 10.0, 0.1)
B_values = [0.0, 0.2, 0.4, 0.6]
N = L * L
spins = np.ones((L, L))
plt.figure(figsize=(12, 6))

# Plotting of Mean Total Energy vs T for different values of B
for B in B_values:
    mean_total_energy_T = []
    for T in temperatures:
        energy_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_2D_nonzero_B(spins, T, B)
            if sweep > relax_sweeps:
                E = energy_ising_2D_nonzero_B(spins, B)
                energy_sweep = np.append(energy_sweep, E)
        meanE = np.mean(energy_sweep)  # Mean energy
        mean_total_energy_T.append (meanE)
    plt.plot(temperatures, mean_total_energy_T, "--o", label=f'B={B}')

# Plotting the graph
plt.xlabel('Temperature')
plt.ylabel('Mean Total Energy')
plt.title('Mean Total Energy vs Temperature for varying external magnetic field B')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 3a Code to produce one graph with four plots of mean total energy against temperature for varying external magnetic fields *B*

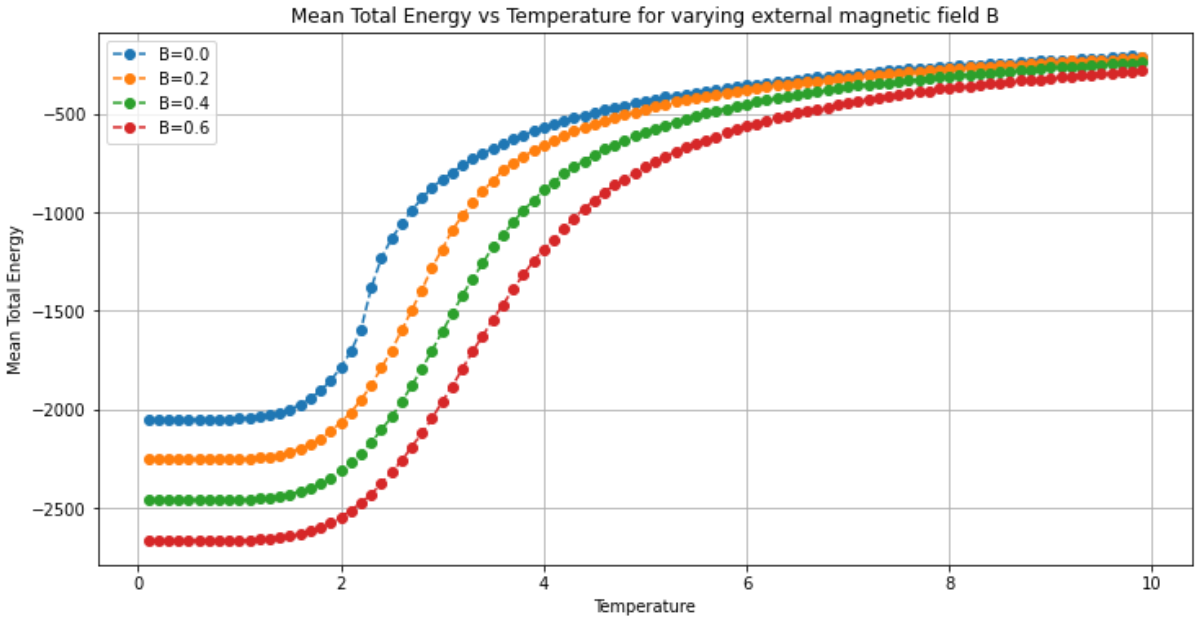


Figure 3b Graph displaying four plots of mean total energy against temperatures from $T = 0.1$ to $T = 10$ when the external magnetic field is $B = 0.0$ (blue curve), $B = 0.2$ (orange curve), $B = 0.4$ (green curve), and $B = 0.6$ (red curve); the size of the lattice is $N = 32 \times 32$

Interpreting the results in Figure 3b, it shows that the external non-zero magnetic field B lowers the energy in comparison to $B = 0$. This was illustrated in Figure 3b, where the curves for higher B values appear lower on the graph, indicating less energy.

According to Subsection 3.1, when the exchange constant is greater than zero, $J > 0$, a pair of spins pointing in the same direction reduces the overall energy of the system. At low temperatures less than $T = 0.2$, the graph with higher B values starts at a lower energy. This is because when a greater non-zero B is applied, more spins are aligned in one direction, that is the direction of B. Therefore, there would be more pairs of spins pointing in the same direction which reduces the overall energy of the system compared to the $B = 0$ case.

After $T = 0.2$, the phase transition can be seen. Above $T = 0.4$, Figure 3b demonstrates that the curves start to converge. This is because thermal energy begins to overcome the effect of the external magnetic field, causing more spins to flip randomly, resulting in a more disordered state. Hence, at higher temperatures, the effect of a non-zero B diminishes, and the mean total energy approaches a value similar to the $B = 0$ case. However, for the temperature range in this investigation, the effect of the magnetic field in reducing the energy compared to $B = 0$ case remains observable.

(iii) Figure 3c presents the code to create a single graph with three plots of specific heat capacity as a function of temperature for varying external magnetic field B. Like part (d)(ii), the function that performs the Monte Carlo sweep on a 2D system of N spins and the function that calculates the energy of the 2D Ising model, both for non-zero B, are used. This is reiterated in Lines 7 to 25 for completeness.

The required parameters were defined in Lines 28 to 35 to produce plots of specific heat capacity as a function of temperature ranging from $T = 0.5$ to $T = 4.0$ for external magnetic field values of $B = 0.0, 0.2, 0.4$. This was demonstrated for a constant lattice size of $N = 64 \times 64$.

Lines 38 to 59 calculated the required values and produced the plots. The specific heat capacity was calculated by first obtaining the mean energy $\langle E \rangle$, similar to part (d)(ii), and the mean square energy $\langle E^2 \rangle$. Both of these values were used in calculating the specific heat capacity c via equation (14) of Unit 10IM:

$$c = \frac{1}{k_B T^2 N} \left(\langle E^2 \rangle - \langle E \rangle^2 \right) .$$

This process is demonstrated in Lines 47 to 49 of Figure 3c.

```
1  %reset -f
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from numba import njit
5
6  @njit
7  def metropolis_sweep_ising_2D_nonzero_B(spins, T, B):
8      L = spins.shape[0]
9      for i in np.random.permutation(L):
10         for j in np.random.permutation(L):
11             DeltaE = 2 * spins[i, j] * (spins[i - 1, j] + spins[(i + 1) % L, j] +
12                                         spins[i, j - 1] + spins[i, (j + 1) % L])\
13             + 2 * spins[i, j] * B
14             if np.exp(-DeltaE / T) > np.random.random():
15                 spins[i, j] = -spins[i, j]
16
17 @njit
18 def energy_ising_2D_nonzero_B(spins, B):
19     L = spins.shape[0]
20     E = 0.0
21     for i in range(L):
22         for j in range(L):
23             E += -spins[i, j] * (spins[i - 1, j] + spins[i, j - 1])\
24             - B * (spins[i, j])
25     return E
26
```

```
# Required parameters
L = 64
relax_sweeps = 500
sweeps = 2000
temperatures = np.arange(0.5, 4.0, 0.1)
B_values = [0.0, 0.2, 0.4]
N = L * L
spins = np.ones((L, L))
plt.figure(figsize=(12, 6))

# Plotting of Specific Heat Capacity vs T for different values of B
for B in B_values:
    specific_heat_T = []
    for T in temperatures:
        energy_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_2D_nonzero_B(spins, T, B)
            if sweep > relax_sweeps:
                E = energy_ising_2D_nonzero_B(spins, B)
                energy_sweep = np.append(energy_sweep, E)
        meanE = np.mean(energy_sweep)  # Mean energy
        meanE2 = np.mean(energy_sweep ** 2)  # Mean square energy
        specific_heat_T.append((meanE2 - meanE ** 2) / (N * T ** 2))
    plt.plot(temperatures, specific_heat_T, "--o", label=f'B={B}')

# Plotting the graph
plt.xlabel('Temperature')
plt.ylabel('Specific Heat Capacity')
plt.title('Specific Heat Capacity vs Temperature for varying '\
          'external magnetic field B')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 3c Code to produce one graph with three plots of the specific heat capacity against temperature for varying external magnetic fields *B*

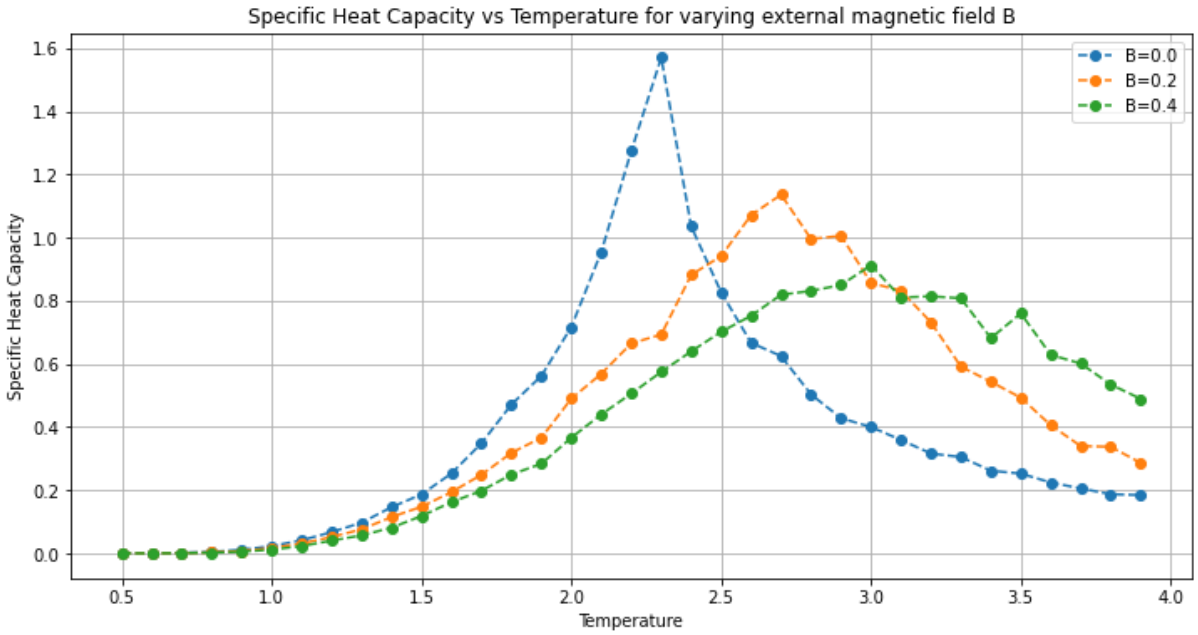


Figure 3d Graph displaying three plots of the specific heat capacity against temperatures from $T = 0.5$ to $T = 4.0$ when the external magnetic field is $B = 0.0$ (blue curve), $B = 0.2$ (orange curve), and $B = 0.4$ (green curve); the size of the lattice is $N = 64 \times 64$

When $B = 0$, Figure 3d shows that the maximum heat capacity is approximately situated at the critical temperature $T_c \approx 2.25$. When the external magnetic field is increased to $B = 0.2$, the maximum heat capacity is less and occurs at a higher temperature compared to $B = 0$; this is shown by the peak of the orange curve that is lower and shifted to the right of the graph compared to the peak of the blue curve. Increasing the external magnetic field to $B = 0.4$, the maximum heat capacity diminishes further and is situated at an even higher temperature than $B = 0.2$ and $B = 0$; this is illustrated in Figure 3d by the peak of the green curve that is lower and shifted more to the right of the graph compared to the peak of the blue and orange curve.

Therefore, these results showed that as the external magnetic field is increased, the maximum heat capacity diminishes. Given that the temperature at which the maximum heat capacity occurs is the critical temperature, increasing B also increases the value of critical temperature, as shown by the shift of the graph peaks to the right.

Part (e)

(i) This part investigated the three-dimensional ferromagnetic Ising model on a cubic lattice to ascertain an approximate value of critical temperature T_c in the infinite system. This was done within two sections: the method section described the code used, and the results section presented the code output along with the analysis discussion.

Method

Throughout this exploration, $J = 1$, $k_B = 1$, and periodic boundary conditions were assumed. This investigation involved the case when there is no external magnetic field, $B = 0$.

The critical temperature T_c was approximated using two methods, both employing Monte Carlo simulations. The first approach involved plotting the magnetisation per spin against temperature for different lattice sizes. Figure 4a shows the code used to perform this task.

From Figure 4a, Lines 8 to 17 show the modified function from Computer session 10IM.1 that performs a Monte Carlo sweep on a 3D system of N spins. The modification includes the addition of the k index that introduces a third dimension to the lattice, requiring an extra loop to iterate over this dimension. The change in energy now accounts for six nearest neighbours in total, compared to four in the two-dimensional case. This function does not include B as it is not a requirement since $B = 0$.

Likewise, Lines 20 to 26 present another modified function from Computer session 10IM.1, which calculates the magnetisation of the 3D system. This modification involves the addition of the k index that introduces a third dimension to the lattice; hence, an extra loop was created to iterate over this dimension.

This simulation allowed the system to relax for 400 sweeps before taking measurements and used 2000 sweeps to estimate average values. The temperature range used for plotting was from $T = 2.0$ to $T = 6.0$. Four plots were created, each corresponding to lattices of size $N = 15 \times 15 \times 15,\ 20 \times 20 \times 20,\ 25 \times 25 \times 25$, and $30 \times 30 \times 30$. These parameters were defined in Lines 29 to 33 of Figure 4a. Lines 36 to 58 calculated the required values and created the plots of magnetisation per spin against temperature. The output is shown in Figure 4c.

```
%%time
%reset -f
import numpy as np
import matplotlib.pyplot as plt
from numba import njit

@njit
def metropolis_sweep_ising_3D(spins, T):
    L = spins.shape[0]
    for i in np.random.permutation(L):
        for j in np.random.permutation(L):
            for k in np.random.permutation(L):
                DeltaE = 2 * spins[i, j, k] * (spins[i - 1, j, k] + spins[(i + 1) % L, j, k] +
                                               spins[i, j - 1, k] + spins[i, (j + 1) % L, k] +
                                               spins[i, j, k - 1] + spins[i, j, (k + 1) % L] )
                if np.exp(-DeltaE / T) > np.random.random():
                    spins[i, j, k] = -spins[i, j, k]

@njit
def magnetic_moment_ising_3D(spins):
    S = 0.0
    for i in range(spins.shape[0]):
        for j in range(spins.shape[0]):
            for k in range(spins.shape[0]):
                S += spins[i, j, k]
    return S

# Required parameters
relax_sweeps = 400
sweeps = 2000
temperatures = np.arange(2, 6, 0.1)
plt.figure(figsize=(12, 6))
L_values = [15, 20, 25, 30]

# Plotting of m vs T for different lattice sizes
for L in L_values:
    N = L * L * L
    magnetisation_T = []
    for T in temperatures:
        spins = np.ones((L, L, L))
        magnetic_moment_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_3D(spins, T)
            magnetic_moment_sweep.append(magnetic_moment_ising_3D(spins) / N)
        magnetisation_T.append(np.mean(magnetic_moment_sweep[relax_sweeps:]))
    plt.plot(temperatures, magnetisation_T, ":o", label=f'N = {L} X {L} X {L}')

# Critical temperature approximate, found via visual inspection
T_c = 4.5

# Plotting the graph
plt.xlabel('Temperature')
plt.ylabel('Magnetisation per spin')
plt.title('Magnetisation per Spin vs Temperature for Different Lattice Sizes')
plt.axvline(T_c, c='k', ls='--', label=r'$T_c$')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 4a Code to produce one graph with four plots of magnetisation per spin against temperature for varying lattice sizes when $B = 0$, for the 3D Ising model

The second approach was plotting the magnetisation susceptibility against temperature for different lattice sizes. Figure 4b shows the code used to perform this.

Lines 8 to 26 of Figure 4b repeated the modified functions from Figure 4a, which are used to perform the Monte Carlo sweep on a 3D system of N spins and calculate the magnetisation of the 3D system. Lines 29 to 33 also specified the same parameters as Figure 4a.

The magnetisation M of the system is equal to the mean total spin $\langle S \rangle$, as stated in equation (11) of Unit 10IM:

$$M = \langle S \rangle .$$

The mean total spin $\langle S \rangle$ and the mean square total spin $\langle S^2 \rangle$ were calculated and used to find the magnetic susceptibility χ, which is given by the equation (13) of Unit 10IM

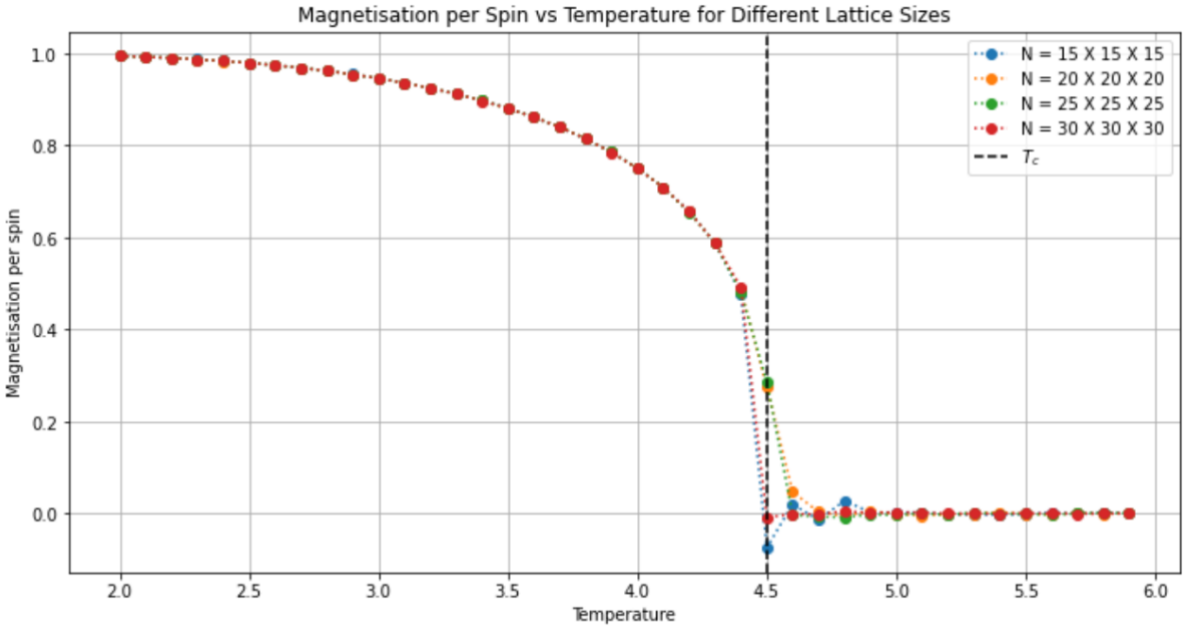
$$\chi = \frac{1}{k_b T N} (\langle S^2 \rangle - \langle S \rangle^2)$$

This process was performed in Lines 45 to 48 of Figure 4b. From Lines 36 to 61, the required values were calculated, and the plots of magnetic susceptibility against temperature were produced. The output is shown in Figure 4d.

```
%%time
%reset -f
import numpy as np
import matplotlib.pyplot as plt
from numba import njit

@njit
def metropolis_sweep_ising_3D(spins, T):
    L = spins.shape[0]
    for i in np.random.permutation(L):
        for j in np.random.permutation(L):
            for k in np.random.permutation(L):
                DeltaE = 2 * spins[i, j, k] * (spins[i - 1, j, k] + spins[(i + 1) % L, j, k] +
                                               spins[i, j - 1, k] + spins[i, (j + 1) % L, k] +
                                               spins[i, j, k - 1] + spins[i, j, (k + 1) % L] )
                if np.exp(-DeltaE / T) > np.random.random():
                    spins[i, j, k] = -spins[i, j, k]

@njit
def magnetic_moment_ising_3D(spins):
    S = 0.0
    for i in range(spins.shape[0]):
        for j in range(spins.shape[0]):
            for k in range(spins.shape[0]):
                S += spins[i, j, k]
    return S

# Required parameters
relax_sweeps = 400
sweeps = 2000
temperatures = np.arange(2, 6, 0.1)
plt.figure(figsize=(12, 6))
L_values = [15, 20, 25, 30]

# Plotting of X vs T for different lattice sizes
for L in L_values:
    N = L * L * L
    magnetic_susceptibility_T = []
    for T in temperatures:
        spins = np.ones((L, L, L))
        total_magnetisation_sweep = []
        total_magnetisation2_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_3D(spins, T)
            total_magnetisation_sweep.append(magnetic_moment_ising_3D(spins))
            total_magnetisation2_sweep.append(magnetic_moment_ising_3D(spins)**2)
        magnetic_susceptibility_T.append((np.mean(total_magnetisation2_sweep[relax_sweeps:])
                                          - np.mean(total_magnetisation_sweep[relax_sweeps:])**2)/(T*L*L*L))
    plt.plot(temperatures, magnetic_susceptibility_T, ":o", label=f'N = {L} X {L} X {L}')

# Critical temperature approximate, found via visual inspection
T_c = 4.5

# Plotting the graph
plt.xlabel('Temperature')
plt.ylabel('Magnetic susceptibility')
plt.title('Magnetic susceptibility vs Temperature for Different Lattice Sizes')
plt.axvline(T_c, c='k', ls='--', label=r'$T_c$')
plt.legend()
plt.grid(True)
plt.show();
```

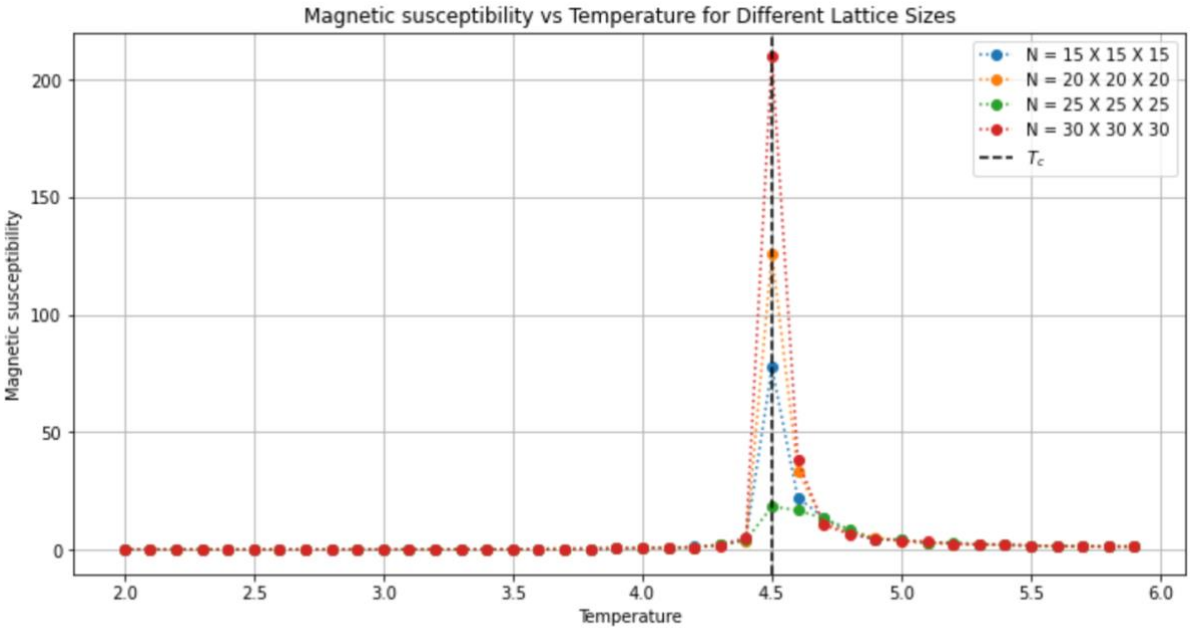
Figure 4b Code to produce one graph with four plots of magnetic susceptibility against temperature for varying lattice sizes when $B = 0$, for the 3D Ising model

Results and interpretation




```
CPU times: total: 58.4 s
```

Figure 4c A single graph presenting four simulation plots of magnetisation per spin as a function of temperature from $T = 2.0$ to $T = 6.0$ (coloured dashed lines) and a line at T_c (dashed black line); the CPU time to produce this graph is 58.4 seconds




```
CPU times: total: 42 s
```

Figure 4d A single graph presenting four simulation plots of magnetic susceptibility as a function of temperature from $T = 2.0$ to $T = 6.0$ (coloured dashed lines) and a line at T_c (dashed black line); the CPU time to produce this graph is 42 seconds

Figure 4c presents four simulation plots of magnetisation per spin against temperatures from $T = 2.0$ to $T = 6.0$. The lattice sizes $15 \times 15 \times 15$, $20 \times 20 \times 20$, $25 \times 25 \times 25$, and $30 \times 30 \times 30$ are represented by the dashed blue, orange, green, and red lines respectively. As temperature increases from $T = 2.0$, the magnetisation per spin of all simulated plots sharply declines to zero, which indicates phase transition. The temperature range where this sharp decline occurs for all lattice sizes is approximately around a single value that is the critical temperature; this can be seen to be approximately $T_c \approx 4.5$. A line at T_c (dashed black line) has been added to the simulated results so that it can be observed more easily. The execution time for the code in Figure 4a was 58.4 seconds, as shown by the CPU time in Figure 4c.

Figure 4d shows four simulation plots of magnetic susceptibility versus temperatures from $T = 2.0$ to $T = 6.0$. The lattice sizes $15 \times 15 \times 15$, $20 \times 20 \times 20$, $25 \times 25 \times 25$, and $30 \times 30 \times 30$ are represented by the dashed blue, orange, green, and red lines respectively. As temperature increases from $T = 2.0$, the magnetic susceptibility of all simulated plots sharply increases to a peak. This means that close to the peak, the small changes in temperature are associated with large changes in the magnetisation of the system if it were subject to an external magnetic field. The temperature where the peaks are situated, that is the critical temperature, for all lattice sizes is approximately $T_c \approx 4.5$. A line at T_c (dashed black line) has been placed on top of the simulated results in Figure 4d. The execution time for the code in Figure 4b was 42 seconds, as shown by the CPU time in Figure 4d.

In the context of an infinite system, as detailed in Computer session 10IM.1, the range in which the magnetisation declines would tend to a single critical temperature with expanding system size. In Figure 4c, increasing lattice size reveals similar decline ranges; hence, if this range were to decrease to a single temperature then $T_c \approx 4.5$ is a good approximation for the critical temperature. Additionally, from Exercise 3 of Computer session 10IM.1, it was stated that in the case of an infinite system, the magnetic susceptibility diverges at the critical temperature, as seen in Figure 4d where the peaks occurred. For all lattice sizes, the temperatures in which the peaks occur are approximately equal- so, for an infinite system, a good approximation for the critical temperature is $T_c \approx 4.5$. In conclusion, this investigation produced an estimate of the critical temperature to be $T_c \approx 4.5$ in the infinite system for the three-dimensional ferromagnetic Ising model.

(ii) From Figure 5a, Lines 8 to 27 show the modified function from Computer session 10IM.1 that performs a Monte Carlo sweep on a 3D system of N spins, as described in part (e)(i), but now considers an external magnetic field B. This was done by adding the term $2s_k B$ for the change in energy, this can be seen in Line 16. This was permitted because the equation for change in energy, that is

$$E_b - E_a = 2s_k \left(J \sum_{i \text{ n.n.of } k} s_i + B \right),$$

applies to an arbitrary number of dimensions, including the three-dimensional case. Likewise, Lines 20 to 27 presented another modified function from Computer session 10IM.1 used to calculate the magnetisation of the 3D system, which remains the same as in part (e)(i).

This simulation allowed the system to relax for 400 sweeps before taking measurements and used 2000 sweeps to estimate average values. The temperature range used for plotting was from $T = 0.2$ to $T = 10.0$. Four plots were created, each corresponding to external magnetic field values $B = 0.005, 0.17, 0.335$, and 0.5. These values of B were chosen to provide evenly spaced intervals from $B = 0.005$ to $B = 0.5$. The lattice size chosen for this task was $N = 20 \times 20 \times 20$ because this was a good compromise between execution time and the quality of plots produced. The parameters were defined in Lines 30 to 36 of Figure 5a.

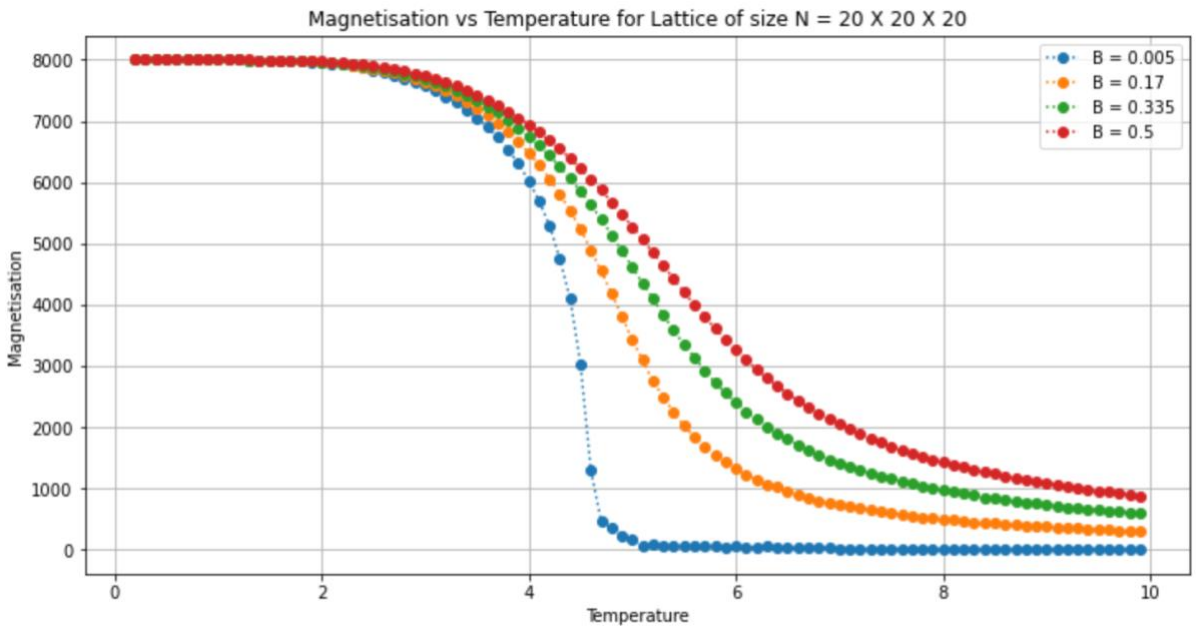
Lines 39 to 56 calculated the required values and created the plots of magnetisation against temperature. Magnetisation M was defined in Subsection 3.2.3 of Unit 10IM, given by equation (11) where

$$M = \langle S \rangle .$$

The code in Lines 41 to 48 of Figure 5a is similar to what was used in part (e)(i) in Figure 4a, where the magnetisation per spin was calculated. The same code was used except for the Monte Carlo sweep which now considers the existence of B and the total magnetisation was not divided by N. The output of the code is shown in Figure 5b.

```
%%time
%reset -f
import numpy as np
import matplotlib.pyplot as plt
from numba import njit

@njit
def metropolis_sweep_ising_3D_nonzero_B(spins, T, B):
    L = spins.shape[0]
    for i in np.random.permutation(L):
        for j in np.random.permutation(L):
            for k in np.random.permutation(L):
                DeltaE = 2 * spins[i, j, k] * (spins[i - 1, j, k] + spins[(i + 1) % L, j, k] +
                                               spins[i, j - 1, k] + spins[i, (j + 1) % L, k] +
                                               spins[i, j, k - 1] + spins[i, j, (k + 1) % L])\
                + 2 * spins[i, j, k] * B
                if np.exp(-DeltaE / T) > np.random.random():
                    spins[i, j, k] = -spins[i, j, k]

@njit
def magnetic_moment_ising_3D(spins):
    S = 0.0
    for i in range(spins.shape[0]):
        for j in range(spins.shape[0]):
            for k in range(spins.shape[0]):
                S += spins[i, j, k]
    return S

# Required parameters
relax_sweeps = 400
sweeps = 2000
temperatures = np.arange(0.2, 10.0, 0.1)
plt.figure(figsize=(12, 6))
L = 20
N = L * L * L
B_values = [0.005, 0.17,  0.335, 0.5]

# Plotting of M vs T for different lattice sizes
for B in B_values:
    total_magnetisation_T = []
    for T in temperatures:
        spins = np.ones((L, L, L))
        magnetic_moment_sweep = []
        for sweep in range(sweeps + relax_sweeps):
            metropolis_sweep_ising_3D_nonzero_B(spins, T, B)
            magnetic_moment_sweep.append(magnetic_moment_ising_3D(spins))
        total_magnetisation_T.append(np.mean(magnetic_moment_sweep[relax_sweeps:]))
    plt.plot(temperatures, total_magnetisation_T, ":o", label=f'B = {B}')

# Plotting the graph
plt.xlabel('Temperature')
plt.ylabel('Magnetisation')
plt.title(f'Magnetisation vs Temperature for Lattice of size N = {L} X {L} X {L}')
plt.legend()
plt.grid(True)
plt.show();
```

Figure 5a Code to produce one graph with four plots of magnetisation against temperature for external magnetic field B when the lattice size is $N = 20 \times 20 \times 20$, for the 3D Ising model




```
CPU times: total: 1min 12s
```

Figure 5b A single graph presenting four simulation plots of magnetisation as a function of temperature from $T = 0.2$ to $T = 10.0$ (coloured dashed lines) for varying external magnetic field of $B = 0.005, 0.17, 0.335, 0.5$; the CPU time to produce this graph was 1 minute and 12 seconds

Figure 5b illustrates the four simulation plots of magnetisation as a function of temperature for the three-dimensional system. For lower temperatures of less than $T = 2.0$, all systems subject to different values of B have constant magnetisation. Beyond $T = 2.0$, starting from the lowest value of external magnetic field $B = 0.005$, there is a sharp decline in the mean total magnetisation that approaches $M = 0$, as shown by the blue curve. However, as B is increased, this decline is less sharp and more gradual. For example, the decline of the $B = 0.335$ case (green curve) is less steeper than the decline of the $B = 0.17$ case (orange curve). When $B = 0.5$ (red curve), the decline is least steepest. In all cases, the curve declines and appears to approach $M = 0$. Interpreting this result, as the external magnetic field increases, the critical temperature T_c increases because the phase transition occurs over a broader range of temperatures.

[*End of paper*]