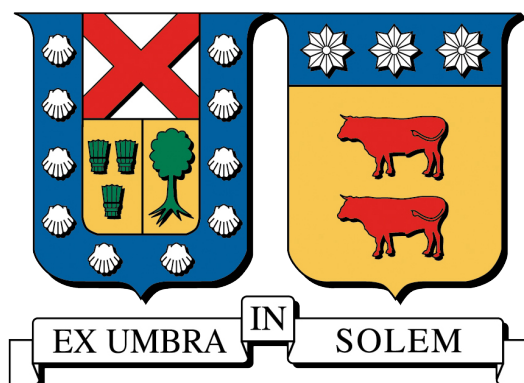


UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA

Departamento de Física

Tesina sometida como requisito para la obtención del grado de Licenciado en Ciencias,
mención Física



Efecto Magnetocalórico en un sistema de q-clock

Tesina escrita por:
Sergio Alejandro Pino Alarcón

Profesor guía:

Prof. Franciso Peña

Comité de Revisión:

Prof. Tomás Corrales

Prof. Pedro Landeros

Prof. Patricio Vargas

Diciembre 2022
Valparaíso, Chile

Resumen

Efecto Magnetocalórico en un sistema de q-clock

Por *Sergio Alejandro Pino Alarcón*

En esta tesina hacemos una revisión bibliográfica sobre el efecto magnetocalórico y también sobre sistemas de espín, para luego obtener resultados sobre el efecto magnetocalórico en un sistema de q -clock. Utilizando la mecánica estadística en conjunto a simulaciones Monte Carlo con algoritmo Metropolis, logramos obtener resultados termodinámicos para redes con q pares entre 2 y 8, de tamaño $L \times L$, con $L = 3, 10, 16, 32, 64, 128$ con condiciones de borde libre, y campos magnéticos que varían entre $B = 0$ y 1 en las unidades naturales del sistema. Las cantidades reportadas por partícula fueron calor específico C/N , Entropía S/N y diferencia de entropía en procesos isotérmicos $\Delta S_{iso}/N$. Encontramos que para $q = 4$ se reporta el mayor peak en C/N , con $B = 1$ y consecuentemente, obteniendo el mayor $-\Delta S_{iso}/N$. Por otro lado, respecto a los efectos del tamaño de red, el efecto es aumentar los valores encontrados en C/N y $-\Delta S_{iso}/N$, pero este efecto cada vez se reduce más hasta no encontrar diferencias significativas entre $L = 64$ y $L = 128$. También, los resultados termodinámicos encontrados están de acorde a los presentados en trabajos anteriores, logrando observar que para $q = 2, 4$ existe solamente una transición de fase de estado ferromagnético a paramagnético, mientras que para $q = 6, 8$ observamos dos transiciones de fase, correspondientes de un estado ferromagnético, a tipo BKT, y finalmente a uno paramagnético.

Agradecimientos

En primer lugar, dar las gracias a mi tutor Francisco Peña por darme la oportunidad de aprender y desarrollarme en este trabajo de investigación, además de brindarme muchísimo apoyo, consejo y enseñarme no solamente en el ámbito académico, si no también como persona. Realmente le estoy eternamente agradecido por todo lo que me ha brindado.

A mi mamá y papá, que a pesar de las diferencias de opiniones y la distancia, me han criado, apoyado y visto salir adelante en mi vida, junto a un amor y cariño que aprecio desde el fondo de mi corazón. A mi hermana, primos/as, en especial a Josefa y Diego, que han sido como una hermana y hermano para mí, muchísimas gracias por todo su apoyo y buenos momentos.

A mis amigos/as, que su apoyo, buenos momentos, cariño y todo lo que me han brindado ha sido crucial en mi vida. Ellos no se imaginan lo tanto que aprecio su compañía y lo tanto que han influido en mí como persona. Muchísimas gracias y los quiero mucho.

Al profesor Patricio Vargas, por la ayuda, tiempo brindado y creación de programas usados en esta investigación, se aprecia todo ello. A Oscar Negrete, que si bien el no sabe, pero una gran parte de esta tesina está basada en la suya, como también su ayuda por los códigos usados.

Finalmente, quiero agradecer el apoyo financiero de los proyectos Fondecyt, Iniciación en Investigación 2020 Grant No. 11200032 y “Millennium Nucleus in NanoBioPhysics” proyecto NNBP #NCN2021_021.

Tabla de contenidos

Resumen	1
Agradecimientos	2
1 Efecto Magnetocalórico	5
§1. Introducción al MCE	5
§2. Termodinámica del MCE	8
§2.1. Ciclo de Carnot versión magnética	10
§3. Resultados del ensemble canónico	11
§3.1. Modelo de Paramagnetismo	12
§3.2. Modelos de Ferromagnetismo	13
§3.2.1. Modelo de Ising en 1 dimensión	13
§3.2.2. Modelo de Ising en campo medio	16
§3.2.3. Modelo de reloj q-Estados (<i>q-State Clock</i>)	21
§4. Simulación Monte-Carlo	22
§4.1. Estimadores	22
§4.2. Muestreo por importancia	23
§4.3. Procesos de Markov	23
§4.4. Tasas de aceptación	25
§4.5. Algoritmo Metropolis	26
§4.6. Mediciones	27
§4.7. Simulación computacional	29
2 Resultados y discusión	30
§1. Resultados analíticos	30
§1.1. Resultados y discusión para 3x3	30
§2. Resultados computacionales	32
§2.1. Resultados	32
§2.1.1. Red 10x10	32
§2.1.2. Red 16x16	34
§2.1.3. Red 32x32	35
§2.1.4. Red 64x64	37
§2.1.5. Red 128x128	38
§2.2. Discusión	40
§3. Conclusiones	41
§4. Proyecciones a futuro	42
3 Anexos	43
§1. Energía por campo externo	43
§2. Ensemble Canónico	45

§3. Códigos computacionales	47
§3.1. Código C++	47
§3.2. Código Mathematica	63
4 Bibliografía	76

Capítulo 1

Efecto Magnetocalórico

§1 Introducción al MCE

El efecto magnetocalórico, o también conocido por sus siglas en inglés, MCE, corresponde a una respuesta térmica de un material magnético ante la presencia de un campo magnético variante. Este efecto es proveniente de los cambios de entropía que sufre el material al tener el acople de un sistema de espín magnético con un campo magnético externo [1].

Este efecto se descubrió por primera vez por E. Warburg, en 1881 [2], y se explicó en la década de 1920 por P. Debye [3] y W.F. Giauque [4], como también se sugirió que serviría para llegar a temperaturas menores a $1K$ desmagnetizando adiabáticamente una sustancia paramagnética, específicamente, primero, teniendo un material expuesto a un gran campo magnético, después enfriar este a la menor temperatura posible por medios convencionales, y luego de esto, aislar térmicamente el material y remover este campo magnético. Esta técnica se usa actualmente en investigación a temperaturas ultrabajas [5].

El interés actual por estudiar este efecto es la refrigeración usando el MCE, o también conocida como refrigeración magnética, la cuál ya es posible realizar a temperatura ambiente, por lo que se trata de aumentar este efecto en ese rango de temperaturas. Refrigeración basada en el MCE tiene las ventajas de que [6]: (i) No es dañina al medioambiente. (ii) Materiales magnéticos tienen densidad entrópica mayor que la de un gas refrigerante. (iii) No necesitan trabajo mecánico para aplicarse. (iv) Eficiencias altas, llegando a un 60% de la del ciclo de Carnot. (v) No necesita mantenimiento continuo. Aún así, todavía existen dificultades que se tienen que superar, específicamente los altos costos, el uso de campos magnéticos fuertes y que el efecto actualmente solo se ha visto con pequeñas variaciones de temperatura. Siguiendo esta idea, el objeto de estudio en este trabajo es el efecto magnetocalórico en un sistema de q -clock, correspondiente a un sistema de espines en donde cada uno tiene q estados posibles, y ver en que condiciones podemos obtener un mayor efecto.

Para poder analizar este efecto, empezamos tratando la entropía total de un material como la suma de tres entropías: electrónica S_e , magnética S_m y de red S_l . [7]. En esta primera representación, vamos a tener que a presión constante P , estas tres entropías dependen de la temperatura T y el campo magnético H :

$$S(T, H) = S_e(T, H) + S_l(T, H) + S_m(T, H) \quad (1.1)$$

En general, se tiene que la entropía magnética depende fuertemente del campo magnético, mientras que es muy común encontrar materiales con entropía electrónica y de red con poca dependencia del campo magnético, pero es posible encontrar materiales que a bajas temperaturas ($\approx < 10$ K) tienen entropía electrónica con una dependencia no lineal en el campo magnético [7]. De todas formas, para poder entender la base del MCE, vamos a asumir que estamos ante un sistema más usual, en

dónde podemos establecer nula dependencia del campo magnético en las entropías mencionadas, con lo que la eq.(1.1) queda de la forma:

$$S(T, H) = S_e(T) + S_l(T) + S_m(T, H) \quad (1.2)$$

Con esto establecido, podemos discutir de forma cualitativa el efecto, que de manera general, se mide con dos parámetros, que son la variación de entropía en un proceso isotérmico ΔS_{iso} , y la variación de temperatura en un proceso adiabático ΔT_{ad} . Comencemos viendo la siguiente figura:

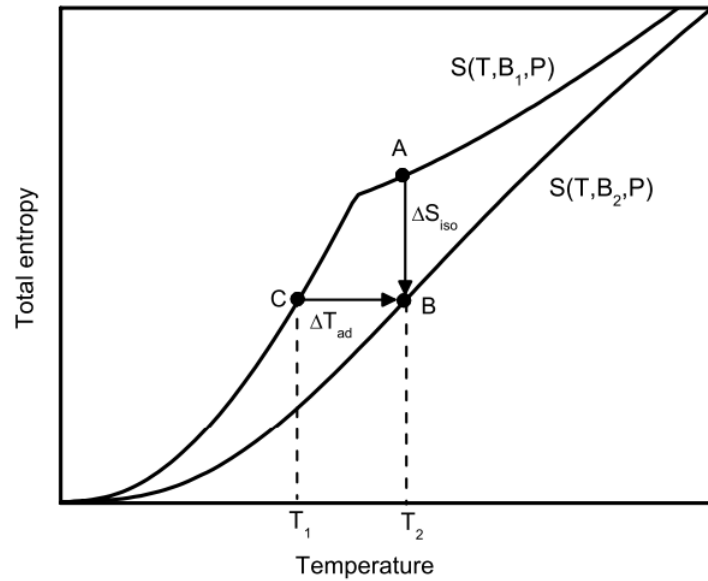


Fig. 1.1: Diagrama entropía versus temperatura para dos campos magnéticos [8].

Entonces, si en un sólido decidimos hacer una variación de campo magnético de H_1 a H_2 en un proceso isotérmico, vemos que nuestra variación de entropía del sistema solamente depende de la variación de la entropía magnética, y esta a su vez del campo magnético, de tal manera que:

$$\Delta S_{iso}(T, \Delta H) = S_m(T, H_2) - S_m(T, H_1) \quad (1.3)$$

También, en un proceso de magnetización adiabática, vamos a tener que el efecto magnetocalórico es una función de la temperatura y la variación de campo magnético, con:

$$\Delta T_{ad}(T, \Delta H) = T_2(S)_{H_2} - T_1(S)_{H_1}, \quad \text{Con } S(H_2, T_2) = S(H_1, T_1) \quad (1.4)$$

Y así vemos las dos formas posibles de medir este efecto [7].

Ahora, para mostrar un ejemplo del efecto y así entender a grandes rasgos que ocurre y su utilidad, vamos a ver que pasa en un ciclo de refrigeración magnética [1]:

Partiendo de un sistema sin un orden magnético dominante, lo sometemos a los siguientes procesos:

1. Le aplicamos rápidamente un campo magnético de tal manera que el proceso sea adiabático, con lo que los espines se reordenan. Al ocurrir esto, notamos que por este reordenamiento la entropía magnética disminuye, y para mantener la entropía del sistema constante, deben aumentar las otras dos entropías, por lo que la temperatura del sistema aumenta.
2. Luego transferimos calor del sistema a un reservorio de menor temperatura, manteniendo el campo constante, y el orden de espines se mantiene.

3. Tras esto, disminuimos el campo magnético rápidamente, o sea, el proceso inverso al hecho inicialmente, por lo tanto la temperatura del sistema disminuye aún más.
4. Se finaliza la refrigeración poniendo en contacto el sistema con la muestra que se quiere enfriar, en dónde se realiza una transferencia de calor desde la muestra al sistema.

Todo este ciclo está representado en la siguiente figura:

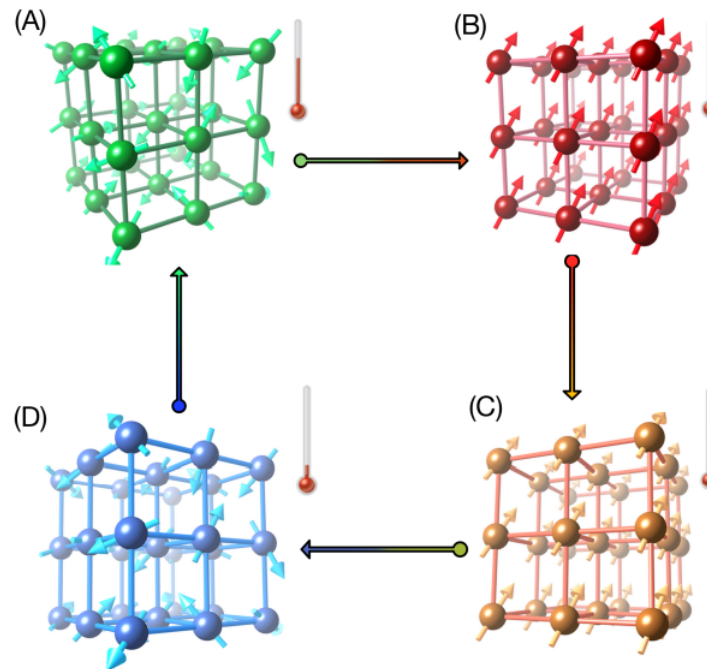


Fig. 1.2: Representación gráfica del proceso de refrigeración magnética [9].

Si bien, no existe un convenio exacto de qué se necesita exactamente para decir que un material óptimo para el efecto, la literatura coincide en varios puntos que se deben necesitar [1]:

1. Temperatura de Debye bajas.
2. Temperatura de Curie cercana a la temperatura ambiente.
3. Altas diferencias de temperaturas (ΔT_{ad}) cerca de las transiciones de fase.
4. Ausencia de histéresis térmica y magnética.
5. Bajo calor específico y alta conductividad térmica.
6. Alta resistencia eléctrica para evitar Corrientes de Foucault al variar el campo magnético.
7. No tóxico.
8. Resistente a la corrosión.
9. Buenas propiedades mecánicas.
10. Bajo coste de producción.
11. Bajo impacto medioambiental.

§2 Termodinámica del MCE

Para poder obtener expresiones necesarias a la hora de describir el MCE, vamos a considerar un sistema no deformable, y bajo la presencia de un campo magnético H , y ocupamos T , P , S , M y V para la temperatura, presión, entropía, magnetización y volumen para cierto estado, respectivamente [10]. Comenzamos estableciendo nuestro diferencial de energía interna dU como ¹:

$$dU(S, V, H) = \delta Q + \delta W = T dS - p dV - H dM \quad (1.5)$$

Haciendo una transformación de Legendre, podemos llegar a la energía libre de Gibbs G :

$$G(T, p, H) = U - TS + pV - MH \quad (1.6)$$

Con lo que su diferencial queda:

$$dG(T, p, H) = V dp - S dT - M dH \quad (1.7)$$

De esto, podemos obtener las siguientes relaciones:

$$V = \left(\frac{\partial G}{\partial p} \right)_{T, H} \quad (1.8)$$

$$S = - \left(\frac{\partial G}{\partial T} \right)_{p, H} \quad (1.9)$$

$$M = - \left(\frac{\partial G}{\partial H} \right)_{T, p} \quad (1.10)$$

Y las siguientes relaciones de Maxwell:

$$\left(\frac{\partial S}{\partial H} \right)_{T, p} = \left(\frac{\partial M}{\partial T} \right)_{p, H} \quad (1.11)$$

$$\left(\frac{\partial S}{\partial p} \right)_{T, H} = - \left(\frac{\partial V}{\partial T} \right)_{p, H} \quad (1.12)$$

$$\left(\frac{\partial S}{\partial M} \right)_{T, p} = - \left(\frac{\partial H}{\partial T} \right)_{M, p} \quad (1.13)$$

Otro parámetro útil para analizar este efecto es la capacidad calórica C_H , el cuál nos permite optimizar que tipo de material transfiere mejor el calor, además de indicar si estamos ante transiciones de fase. Dado que estamos considerando un sistema a presión constante, vamos a tener que:

$$C_{H, p} = \left(\frac{\delta Q}{dT} \right)_{p, H} = T \left(\frac{\partial S}{\partial T} \right)_{p, H} \quad (1.14)$$

Podrán ir notando que estamos dando un gran énfasis a fórmulas relacionadas a la entropía, esto ya que esta cantidad la podemos calcular de distintos modelos y ocupando herramientas de ensembles termodinámicos.

Ahora, dado que estamos trabajando en un proceso sin deformaciones, o sea, a presión constante, vamos a establecer nuestro diferencial de entropía como función de las cantidades T y H :

$$dS(T, H) = \left(\frac{\partial S}{\partial T} \right)_H dT + \left(\frac{\partial S}{\partial H} \right)_T dH \quad (1.15)$$

¹Notamos que la constante μ_0 se iguala a 1.

Ocupando las ec. 1.11 y 1.14, este queda como:

$$dS(T, H) = \frac{C_H}{T} dT + \left(\frac{\partial M}{\partial T} \right)_H dH \quad (1.16)$$

Y de esta ecuación, podemos obtener el cambio de temperatura en un proceso adiabático como:

$$\begin{aligned} 0 &= \frac{C_H}{T} dT + \left(\frac{\partial M}{\partial T} \right)_H dH \\ \Delta T_{ad} &= - \int_{H_1}^{H_2} \frac{T}{C_H} \left(\frac{\partial M}{\partial T} \right)_H dH \end{aligned} \quad (1.17)$$

Para procesos isotérmicos, ocupando 1.16, podemos calcular el cambio de entropía como:

$$\Delta S_{iso} = \int_{H_1}^{H_2} \left(\frac{\partial M}{\partial T} \right)_H dH \quad (1.18)$$

Y en procesos con campo magnético constante, tenemos que el cambio de entropía es:

$$(\Delta S)_H = \int_{T_1}^{T_2} \frac{C_H}{T} dT \quad (1.19)$$

Observamos que, para cambios de T y C_H pequeños, la eq. 1.17 se puede aproximar a [7]:

$$\begin{aligned} \Delta T_{ad} &\approx - \frac{T}{C_H} \int_{H_1}^{H_2} \left(\frac{\partial M}{\partial T} \right)_H dH \\ \Delta T_{ad} &\approx - \frac{T}{C_H} \Delta S_{iso} \end{aligned} \quad (1.20)$$

Con lo que nos enfocamos en calcular $-\Delta S_{iso}$ para poder reportar el efecto.

Igualmente es importante recordar la energía libre de Helmholtz y su forma diferencial para varios cálculos que se realizarán [11]:

$$F(T, V, H) = U - TS \quad (1.21)$$

$$dF(T, V, H) = -S dT - p dV - M dH \quad (1.22)$$

Con lo que podemos obtener:

$$S = - \left(\frac{\partial F}{\partial T} \right)_{V, H} \quad (1.23)$$

$$M = - \left(\frac{\partial F}{\partial H} \right)_{V, T} \quad (1.24)$$

Así, de las ecuaciones anteriores, la energía interna puede ser reescrita como:

$$U = TS + F = F - T \left(\frac{\partial F}{\partial T} \right)_{V, H} = -T^2 \frac{\partial}{\partial T} \left(\frac{F}{T} \right) = \frac{\partial(F/T)}{\partial(1/T)} \quad (1.25)$$

§2.1 Ciclo de Carnot versión magnética

Como se ha dicutado anteriormente, una de las motivaciones para estudiar el MCE es poder encontrar alternativas mejores a mecanismos de refrigeración usados actualmente, que están basados en compresión y expansión de gases, con lo cuál, se pueden liberar gases dañinos al medioambiente. Los ciclos térmicos usados en estos mecanismos hacen uso de cambios de entropía para poder realizar su efecto, y en el caso del MCE, hacemos cambios de temperatura y entropía a través de cambios de un campo magnético, con lo que podemos establecer que todos los procesos ocupados en las máquinas térmicas de gases, pueden ser realizados a través del MCE [12].

En general, los refrigeradores magnéticos se componen de: Un material magnético, un sistema magnetizante/demagnetizante, intercambiadores de calor caliente y frios, y un sistema de transferencia de calor junto a un fluido térmico, encargándose de transferir el calor desde el material y los intercambiadores de calor.

Para ver un ejemplo de un ciclo ideal, procedemos a estudiar el ciclo de Carnot en su versión magnética, el cuál está compuesto de:

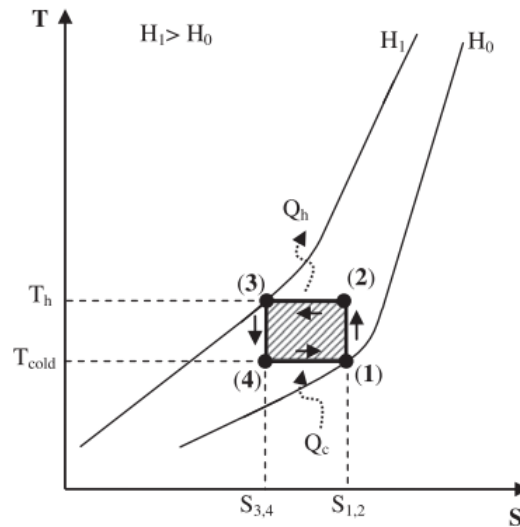


Fig. 1.3: Diagrama temperatura versus entropía para un ciclo de Carnot magnético [1].

1. Comenzamos de un estado (T_c, S_1, H_1) , aplicamos adiabáticamente un campo externo H_2 . Ocupando los argumentos de la parte 1 para explicar la refrigeración magnética, como debe disminuir S_m , aumenta S_e y S_l lo que conlleva a un aumento de temperatura a T_h .
2. Luego, isotérmicamente aumentamos el campo magnético a H_3 , llegando a la configuración (T_h, S_2, H_3) , proceso en el cuál el sistema entrega calor Q_h .
3. Disminuimos el campo magnético a H_4 adiabáticamente, en dónde igualmente la temperatura se disminuye hasta T_c .
4. Isotérmicamente disminuimos el campo magnético hasta llegar a la configuración inicial (T_c, S_1, H_1) , proceso en el cuál el sistema recibe calor Q_c .

Notamos que los procesos que lo componen, justamente son los más importantes para el MCE, el proceso isotérmico y adiabático.

Ahora, podríamos pensar que este es un buen ciclo a replicar en general, pero resulta que el ciclo, al ocupar campos magnéticos variables en cada uno de los puntos de trabajo, necesita un electroimán

o un imán superconductor para que el campo magnético sea controlado, los cuáles son ineficientes en su uso de energías, lo cuál convierte en el ciclo en uno no apto para la refrigeración.

También, existen otros tipos de ciclos que son ocupados mayormente, en especial la regeneración magnética activa (Active magnetic regenerator cycle o AMR) [1], pero su estudio está fuera del propósito de este trabajo.

§3 Resultados del ensemble canónico

Si bien, todos los resultados anteriores son correctos y útiles, en términos de lo que vamos a estudiar son insuficientes, y en general, nos valemos de la teoría de ensembles, específicamente, del ensemble canónico, para poder obtener las cantidades que nos interesan.

Para esto, realizaremos un pequeño resumen conceptual y de fórmulas importantes para este ensemble [11]².

Primero, la función de partición se puede expresar como:

$$Z(T, B) = \sum_n g_n e^{-\beta E_i}, \quad \text{Con } \beta = 1/(k_B T) \quad (1.26)$$

Con esta obtenida, las otras cantidades termodinámicas pueden ser obtenidas con:

$$F(T, B) = -k_B T \ln Z(T, B) \quad (1.27)$$

$$S(T, B) = \frac{\partial}{\partial T} (k_B T \ln Z(T, B)) = \frac{U}{T} + k_B \ln Z(T, B) \quad (1.28)$$

$$U(T, B) = \langle E \rangle = -\frac{\partial \ln Z}{\partial \beta} = k_B T^2 \frac{\partial}{\partial T} \ln Z(T, B) \quad (1.29)$$

$$C(T, B) = \frac{\partial U}{\partial T} \quad (1.30)$$

Otra relación que se usará es la siguiente:

$$C = k_B \beta^2 (\langle E^2 \rangle - \langle E \rangle^2) \quad (1.31)$$

Deducida de:

$$\begin{aligned} \langle E^2 \rangle - \langle E \rangle^2 &= \frac{1}{Z} \frac{\partial^2 Z}{\partial \beta^2} - \left(\frac{1}{Z} \frac{\partial Z}{\partial \beta} \right)^2 = \frac{\partial}{\partial \beta} \left(\frac{1}{Z} \frac{\partial Z}{\partial \beta} \right) \\ &= \frac{\partial^2 \ln Z}{\partial \beta^2} \\ &= -\frac{\partial U}{\partial \beta} = -\frac{1}{k_B} \frac{\partial U}{\partial (1/T)} \\ &= -\frac{1}{k_B} \left(\frac{\partial T^{-1}}{\partial T} \right)^{-1} \frac{\partial U}{\partial T} = \frac{T^2}{k_B} C \\ &= \frac{C}{k_B \beta^2} \end{aligned}$$

²Si desea revisar con mayor detalle el tema, puede revisar el libro *Statistical Mechanics* de R. K. Pathria y P. D. Beale.

§3.1 Modelo de Paramagnetismo

Primero, estudiaremos lo que pasa en un sistema de N dipolos magnéticos, cada uno con momento magnético μ y un campo magnético externo $\mathbf{H}$. En general, esperamos que el sistema de dipolos se alinee con este campo magnético, pero no en su totalidad, ya que la temperatura hace un efecto de resistir a esta tendencia. También, para temperaturas $T \rightarrow 0$, esperamos que no exista esta resistencia por parte de la temperatura, y para $T \rightarrow \infty$, haya una magnetización nula, dado que los dipolos deberían estar completamente desalineados unos de otros.

Primero, la energía de este sistema es:

$$E_{P.M.} = - \sum_{i=1}^N \mu_i \cdot \mathbf{H} = -\mu H \sum_{i=1}^N \cos \theta_i \quad (1.32)$$

De esto, la función de partición es:

$$\begin{aligned} Z(T, H, N) &= [Z(T, H, 1)]^N = \left[\int_0^{2\pi} d\phi \int_{-1}^1 d(\cos \theta) e^{\beta \mu H \cos \theta} \right]^N \\ Z(T, H, N) &= \left(4\pi \frac{\sinh(\beta \mu H)}{\beta \mu H} \right)^N \end{aligned} \quad (1.33)$$

Y el potencial de Helmholtz:

$$F(T, H, N) = -Nk_B T \ln \left(4\pi \frac{\sinh(\beta \mu H)}{\beta \mu H} \right) \quad (1.34)$$

Y ocupando 1.24, se obtendrá que:

$$M = N\mu \left[\coth \beta \mu H - \frac{1}{\beta \mu H} \right] = N\mu L(\beta \mu H), \quad (1.35)$$

con $L(x)$ la función de Langevin:

$$L(x) = \coth x - \frac{1}{x} \quad (1.36)$$

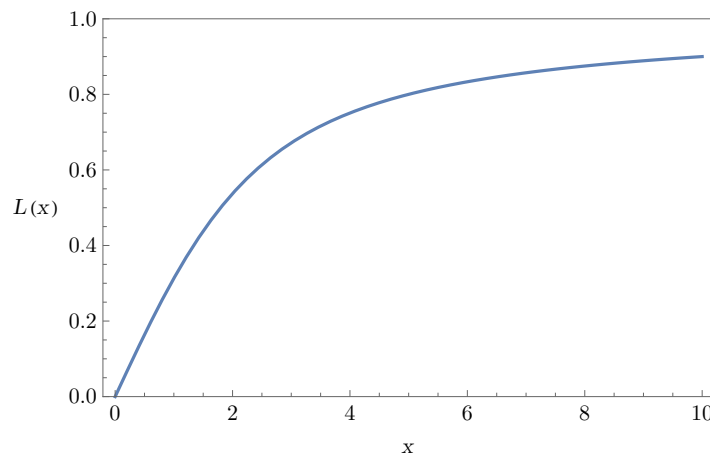


Fig. 1.4: Gráfica de la función de Langevin.

Viendo una gráfica de esta función, como en la figura anterior, notamos que se cumple lo esperado, ya que a campos fuertes $H \gg 1$ o ($T \ll 1$), la función de Langevin tiende a 1, y para $T \gg 1$ (o $H \ll 1$), la función de Langevin tiende a ser lineal en su argumento, y tiene la forma $x/3 - x^3/45 + \dots$, con lo que la magnetización tiende a un valor pequeño.

Finalmente, las cantidades termodinámicas, ocupando (1.28), (1.29) y (1.30), son:

$$S = Nk_B \ln \left(4\pi \frac{\sinh(\beta\mu H)}{\beta\mu H} \right) - \frac{N\mu H}{T} L(\beta\mu H) \quad (1.37)$$

$$U = F + TS = -N\mu H L(\beta\mu H) \quad (1.38)$$

$$C = \frac{Nk_B}{H} \left[1 - \left(\frac{\beta\mu H}{\sinh(\beta\mu H)} \right)^2 \right] \quad (1.39)$$

§3.2 Modelos de Ferromagnetismo

En los modelos de Ferromagnetismo, en general estamos analizando qué es lo que le pasa a una red de espines (o momentos magnéticos) al aumentar la temperatura del sistema, o aplicar un campo magnético externo, etc. También, tenemos que considerar que tenemos varias formas de escribir que pasa en términos de interacciones de los espines [13].

Para poder ver los procedimientos para poder analizar estos sistemas, vamos a realizar algunos cálculos en unos modelos simplificados, para luego, entrar en la discusión del modelo que queremos tratar.

§3.2.1 Modelo de Ising en 1 dimensión

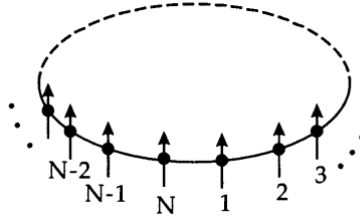


Fig. 1.5: Cadena cerrada de una red de Ising 1D [13].

Para este modelo, consideramos que tenemos una cadena cerrada de espines, en la cuál tenemos dos tipos de interacción: entre espines, que depende de la orientación relativa entre espines vecinos (nearest neighbours, n.n.), y la interacción con un campo magnético externo, en dónde esperamos que cada espín se intente alinear con este. Con esto, el Hamiltoniano de este modelo (caso general de Ising) es el siguiente ³:

$$H_{Ising} = -J \sum_{\langle i,j \rangle} \sigma_i \sigma_j - \mu B \sum_{i=1}^N \sigma_i, \quad \sigma_i = \pm 1 \quad (1.40)$$

El origen del primer término es proveniente de una interacción de exchange entre espines, de origen totalmente cuántico, dado que un tratamiento clásico para poder explicar el fenómeno de alineamiento paralelo (o el caso de alinearse anti paralelamente) no es lo suficientemente fuerte

³De acá en adelante, cambiaremos la notación de campo externo H con B para evitar confusiones con el Hamiltoniano.

para explicar ciertos fenómenos. En general, tenemos que la interacción de exchange proviene de la antisimetrización de la función de onda de dos partículas, con interacción:

$$J_{nm} = \int \psi_m^*(\mathbf{r}_1) \psi_n^*(\mathbf{r}_2) U_{nm} \psi_m(\mathbf{r}_2) \psi_n(\mathbf{r}_1) d^3r_1 d^3r_2 \quad (1.41)$$

Y la diferencia de energía entre configuraciones con espines paralelos y antiparalelos es:

$$\epsilon_{\uparrow\uparrow} - \epsilon_{\downarrow\uparrow} = -2J_{nm} \quad (1.42)$$

De lo que vemos que si $J_{nm} > 0$ (Caso ferromagnético), se prefieren estados con espines alineados.

Ahora, volviendo al análisis del modelo, notamos que podemos reescribir el Hamiltoniano de una forma más simétrica, recordando que estamos en una cadena cerrada, y periódica, con lo que el vecino más cercano al espín σ_i es $\sigma_{i\pm 1}$. Notar que para evitar contar dos veces la misma interacción solo nos interesa el σ_{i+1} . Así, reordenando la segunda suma, el Hamiltoniano toma la forma:

$$H_{Ising} = -J \sum_{i=1}^N \sigma_i \sigma_{i+1} - \frac{1}{2} \mu B \sum_{i=1}^N (\sigma_i + \sigma_{i+1}) \quad (1.43)$$

En este caso, nuestra función de partición es:

$$Z(T, B) = \sum_{\sigma_1=\pm 1} \dots \sum_{\sigma_N=\pm 1} \exp \left[\beta \sum_i^N \left\{ J \sigma_i \sigma_{i+1} + \frac{1}{2} \mu B (\sigma_i + \sigma_{i+1}) \right\} \right] \quad (1.44)$$

Definimos el operador P en el espacio de espines 2 x 2 con sus elementos de matriz:

$$\langle \sigma_i | P | \sigma_{i+1} \rangle = \exp \left[\beta \sum_i^N \left\{ J \sigma_i \sigma_{i+1} + \frac{1}{2} \mu B (\sigma_i + \sigma_{i+1}) \right\} \right] \quad (1.45)$$

Asignando el $\sigma_i = +1$ como $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ y $\sigma_i = -1$ como $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$, tenemos que:

$$P = \begin{pmatrix} e^{\beta(J+\mu B)} & e^{-\beta J} \\ e^{-\beta J} & e^{\beta(J-\mu B)} \end{pmatrix} \quad (1.46)$$

Ocupando la relación de clausura $\sum_{\sigma=\pm 1} |\sigma\rangle\langle\sigma| = 1$, y que $\sigma_{N+1} = \sigma_1$, reemplazamos esto en la función de partición:

$$Z(T, B) = \sum_{\sigma_1=\pm 1} \dots \sum_{\sigma_N=\pm 1} \langle \sigma_1 | P | \sigma_2 \rangle \langle \sigma_2 | P | \sigma_3 \rangle \dots \langle \sigma_N | P | \sigma_{N+1} \rangle \quad (1.47)$$

$$Z(T, B) = \sum_{\sigma_1=\pm 1} \langle \sigma_1 | P^N | \sigma_1 \rangle = \text{Tr } P^N = \lambda_1^N + \lambda_2^N \quad (1.48)$$

con λ_1 y λ_2 los autovalores del operador P , que son:

$$\lambda_{1,2} = e^{\beta J} \cosh(\beta \mu B) \pm \left\{ e^{-2\beta J} + e^{2\beta J} \sinh^2(\beta \mu B) \right\}^{1/2} \quad (1.49)$$

Como comentario, este método de resolución ocupa la llamada matriz de transferencia, la que nos permite realizar un cálculo rápido y relativamente fácil de Z .

Ahora, volviendo a analizar Z , notamos que $\lambda_1 > \lambda_2$, con lo que para $N \rightarrow \infty$ (Límite termodinámico), podemos hacer la siguiente aproximación:

$$Z(T, B) = \lambda_1^N + \lambda_2^N = \lambda_1^N \left(1 + \frac{\lambda_2^N}{\lambda_1^N} \right) \approx \lambda_1^N \quad (1.50)$$

Entonces, ocupando lo anterior y la ecuación (1.27), la energía libre de Helmholtz se puede aproximar a:

$$F(T, B) \approx -Nk_B T \ln \left(e^{\beta J} \cosh(\beta \mu B) + \left\{ e^{-2\beta J} + e^{2\beta J} \sinh^2(\beta \mu B) \right\} \right) \quad (1.51)$$

Una cantidad que puede ser importante a la hora de analizar las transiciones de fase del sistema es la magnetización M , que la podemos calcular de (1.24):

$$M(T, B) = \frac{e^{2\beta J} N \mu \sinh(\beta \mu B)}{(1 + e^{4\beta J} \sinh^2(\beta \mu B))^{1/2}} \quad (1.52)$$

La cuál notamos que para $B = 0$, $M = 0$, lo que nos dice que es imposible tener una magnetización espontánea a cualquier temperatura $T \neq 0$, pero, si tomamos $T \rightarrow 0$, con cualquier $B \neq 0$, tenemos que la magnetización tiende a $N\mu$, que es la mayor magnetización que puede tener el sistema, todos los espines están alineados, y por lo tanto, con esto podemos demostrar que debe existir una transición de fase a un comportamiento ferromagnético en este modelo, en $T_c = 0$. También, ocupando las ecuaciones (1.28), (1.29), (1.30), podemos obtener las gráficas de las cantidades termodinámicas por partícula $S(T, B)/N$; $U(T, B)/N$; $C(T, B)/N$ presentadas a continuación:

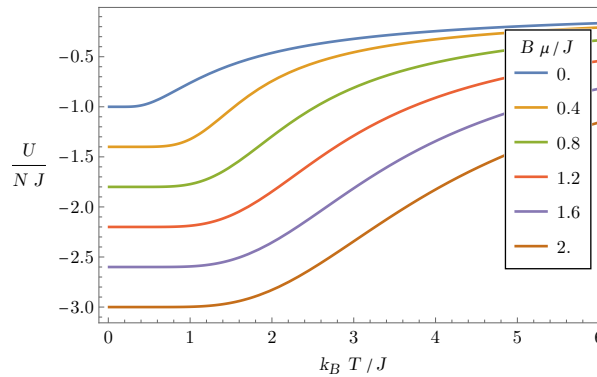


Fig. 1.6: Gráfico Energía interna versus temperatura para el modelo de Ising 1D.

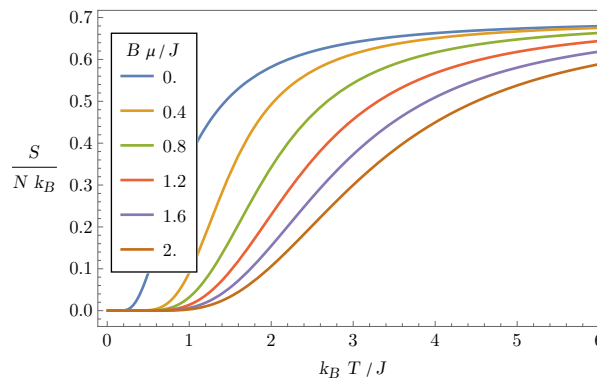


Fig. 1.7: Gráfico Entropía versus temperatura para el modelo de Ising 1D.

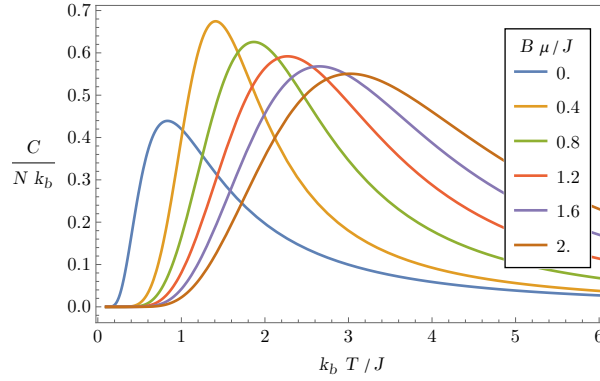


Fig. 1.8: Gráfico Calor específico a campo constante versus temperatura para el modelo de Ising 1D.

Igualmente podemos estudiar el efecto magnetocalórico al obtener ΔS :

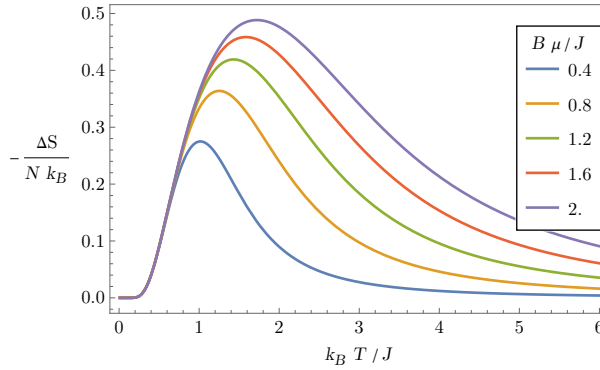


Fig. 1.9: Gráfico ΔS versus temperatura para el modelo de Ising 1D. Cabe destacar que no obtuvimos la gráfica para $B = 0$ dado que es la nula ($\Delta S = 0$).

§3.2.2 Modelo de Ising en campo medio

Hasta ahora, lo que hemos considerado es que cada espín interactúa con sus vecinos más cercanos y un campo externo, pero resulta que solamente existen soluciones analíticas para el caso 1D y el caso 2D sin campo externo (Solución de Onsager). Por lo tanto, lo que se realiza es una aproximación al modelo de Ising (en cualquier dimensión) para poder obtener resultados.

Esta aproximación se realiza intercambiando la interacción con los vecinos más cercanos, a una interacción con el espín promedio de la red, y en términos matemáticos lo que hacemos es ignorar las fluctuaciones de espín, para lo cual vemos la siguiente identidad [14]:

$$\sigma_i \sigma_j = \sigma_i \langle \sigma_j \rangle + \sigma_j \langle \sigma_i \rangle - \langle \sigma_i \rangle \langle \sigma_j \rangle + (\sigma_i - \langle \sigma_i \rangle)(\sigma_j - \langle \sigma_j \rangle) \quad (1.53)$$

Notamos que los valores promedios de los espines son iguales para cada uno ya que el sistema presenta una invariancia traslacional, y que el último término son las fluctuaciones de espín con respecto a su promedio, el cuál vamos a despreciar. Con esto, el Hamiltoniano de Ising se convierte en:

$$H_{Ising} = -J \sum_{\langle i,j \rangle} \{ \sigma_i \langle \sigma \rangle + \sigma_j \langle \sigma \rangle - \langle \sigma \rangle \langle \sigma \rangle \} - \mu B \sum_{i=1}^N \sigma_i \quad (1.54)$$

Los primeros dos términos, por la simetría traslacional del sistema, son equivalentes, y la suma por los vecinos cercanos la podemos escribir como:

$$\sum_{\langle i,j \rangle} = \frac{1}{2} \sum_{i=1}^N \sum_{\langle i \rangle} = \frac{Q}{2} \sum_{i=1}^N, \quad (1.55)$$

en dónde lo que hicimos fue cambiar la suma por vecinos cercanos, por una en dónde vemos los vecinos cercanos de cada sitio, denotada por $\langle i \rangle$ y agregamos un factor de $1/2$ para evitar el doble conteo por sitio, y dado que ya no nos interesa la interacción entre vecinos, podemos cambiar esta suma por un número Q que es llamado número de coordinación que nos indica el número de vecinos de cada sitio, el cuál es dependiente de la geometría que analicemos, por ejemplo, para una red 1D, sería $Q = 2$, para una red 2D cuadrada, sería $Q = 4$.

Con estas simplificaciones, el Hamiltoniano anterior (que renombraremos a Hamiltoniano de campo medio $H_{m.f.}$) nos queda como:

$$H_{m.f.} = \frac{NQJ \langle \sigma \rangle^2}{2} - \sum_{i=1}^N \{JQ \langle \sigma \rangle \sigma_i + \mu B \sigma_i\} \quad (1.56)$$

$$= \frac{NQJ \langle \sigma \rangle^2}{2} - \mu(B_{m.f.} + B) \sum_{i=1}^N \sigma_i \quad (1.57)$$

En dónde el término $B^{m.f.}$ es un campo magnético producido por los espines, y tiene la forma:

$$B^{m.f.} = \frac{QJ \langle \sigma \rangle}{\mu} \quad (1.58)$$

Y podemos definir un campo magnético efectivo de la forma:

$$B_{eff} = B_{m.f.} + B \quad (1.59)$$

Luego, la función de partición del sistema la calculamos con (1.26) junto a (1.57), y nos queda:

$$\begin{aligned} Z_{m.f.}(T, B) &= \sum_{\sigma_1=\pm 1} \dots \sum_{\sigma_N=\pm 1} e^{-\beta N Q J \langle \sigma \rangle^2 / 2} e^{-\beta \mu B_{eff} \sum_{i=1}^N \sigma_i} \\ &= e^{-\beta N Q J \langle \sigma \rangle^2 / 2} \sum_{\sigma_1=\pm 1} \dots \sum_{\sigma_N=\pm 1} e^{-\beta \mu B_{eff} \sum_{i=1}^N \sigma_i} \\ &= e^{-\beta N Q J \langle \sigma \rangle^2 / 2} \left(\sum_{\sigma=\pm 1} e^{-\beta \mu B_{eff} \sigma} \right)^N \\ &= e^{-\beta N Q J \langle \sigma \rangle^2 / 2} \left(e^{-\beta \mu B_{eff}} + e^{\beta \mu B_{eff}} \right)^N \\ Z_{m.f.}(T, B) &= e^{-\beta N Q J \langle \sigma \rangle^2 / 2} [2 \cosh(\beta \mu B_{eff})]^N \end{aligned} \quad (1.60)$$

Ahora, tenemos que la función de partición depende del valor promedio de los espines, pero en general, cantidades promedio las podemos obtener ocupando la función de partición, y en este caso,

ocupando la siguiente fórmula:

$$\langle \sigma \rangle = \frac{1}{N} \left\langle \sum_{i=1}^N \sigma_i \right\rangle = \frac{1}{N} \frac{\sum_{i=1}^N \sigma_i e^{-\beta H_{m.f.}}}{Z_{m.f.}} \quad (1.61)$$

$$\begin{aligned} &= \frac{1}{N Z_{m.f.}} \left(\sum_{i=1}^N \frac{\partial}{\partial (\beta \mu B_{eff})} e^{-\beta H_{m.f.}} \right) \\ &= \frac{1}{\beta \mu N Z_{m.f.}} \frac{\partial Z_{m.f.}}{\partial B_{eff}} \\ \langle \sigma \rangle &= \frac{1}{\beta \mu N} \frac{\partial \ln Z_{m.f.}}{\partial B_{eff}} \end{aligned} \quad (1.62)$$

Con lo que obtenemos:

$$\begin{aligned} \langle \sigma \rangle &= \frac{1}{\beta \mu N} \frac{\partial}{\partial B_{eff}} \left(\ln \left(e^{-\beta N Q J \langle \sigma \rangle^2 / 2} 2^N \right) + N \ln \cosh (\beta \mu B_{eff}) \right) \\ \langle \sigma \rangle &= \tanh (\beta \mu B_{eff}) = \tanh (\beta Q J \langle \sigma \rangle + \beta \mu B), \end{aligned} \quad (1.63)$$

que corresponde a una ecuación autoconsistente (y trascendental) para determinar $\langle \sigma \rangle$, del cuál podemos determinar la magnetización promedio $\langle M \rangle$, notando que esta es:

$$\langle M \rangle = N \mu \langle \sigma \rangle \quad (1.64)$$

Para analizar esto, primero vamos a ver el caso con $B = 0$, en dónde podemos intentar obtener una solución gráfica para $\langle \sigma \rangle$, en la figuras siguientes:

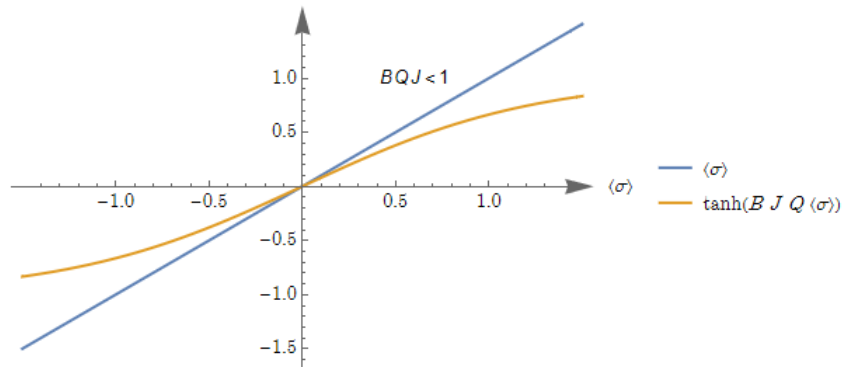
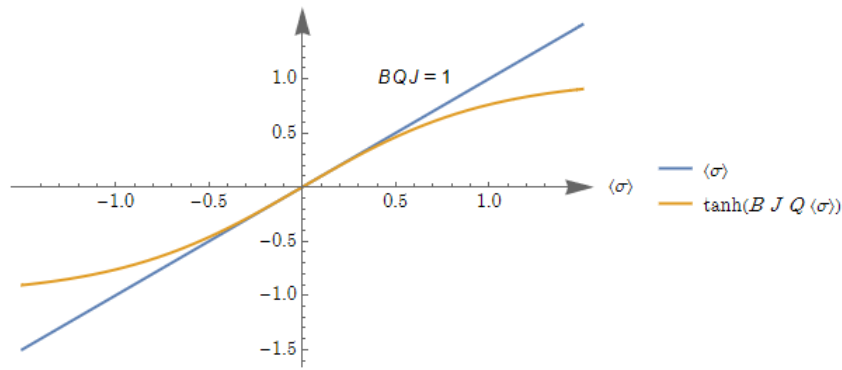
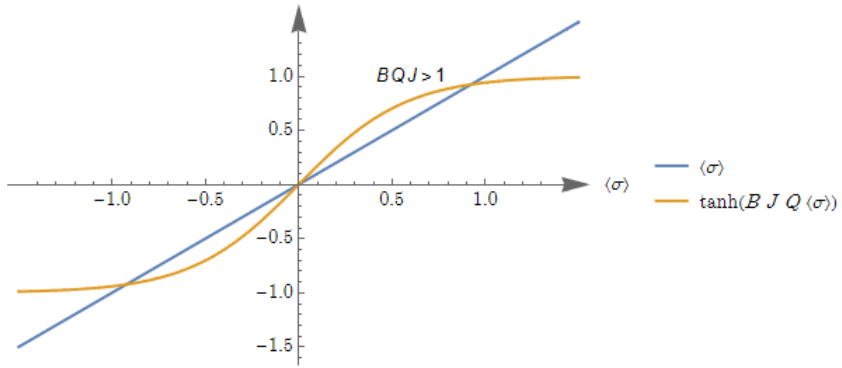


Fig. 1.10: Gráfica para ambos lados de la ecuación autoconsistente, con $\beta Q J < 1$.

Fig. 1.11: Gráfica para ambos lados de la ecuación autoconsistente, con $\beta QJ = 1$.Fig. 1.12: Gráfica para ambos lados de la ecuación autoconsistente, con $\beta QJ > 1$.

Vemos que hay soluciones cualitativamente diferentes con $\beta QJ \leq 1$ (Con $T \geq QJ/k_b$) o $\beta QJ > 1$ (Con $T < QJ/k_b$), siendo el primer caso un régimen paramagnético, con $\langle M \rangle = 0$, y el segundo el régimen ferromagnético $\langle M \rangle = \pm M_0$ ó 0. Por esta razón, dejamos que nuestra temperatura de transición en este modelo (a campo nulo) sea:

$$T_c = \frac{QJ}{k_b} \quad (1.65)$$

Comparando con el modelo de Ising 1D, correspondiente a campo medio con $Q = 2$, este resultado es incorrecto, aunque ocurre que, en general, para altos números de coordinación, este es una buena aproximación, e incluso en el límite de $Q \rightarrow \infty$, este resultado debería ser exacto [15].

También, podemos intentar obtener una gráfica de la solución positiva de $\langle \sigma \rangle$, en dónde podemos ver más detalladamente cuál es el comportamiento de este al aumentar la temperatura:

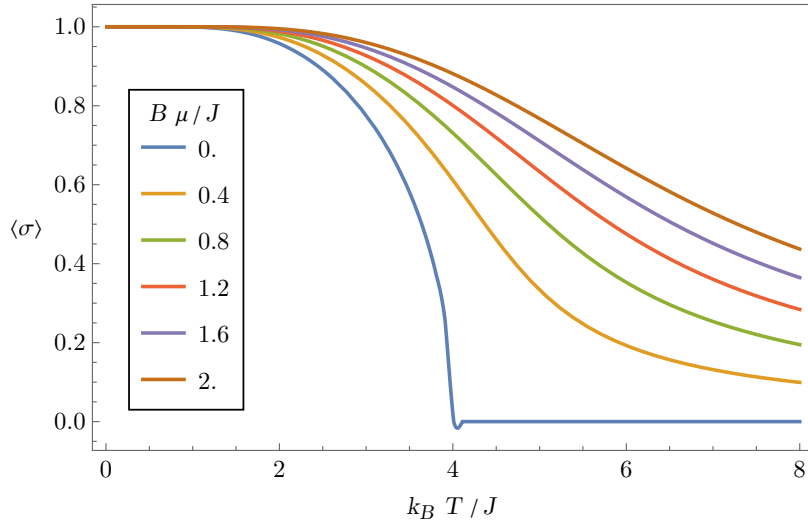


Fig. 1.13: Gráfica de $\langle \sigma \rangle$ versus T para distintos valores de campo magnético, con $Q = 4$, obtenida numéricamente.

Igualmente, se puede demostrar que el sistema no asume cualquiera de las posibles magnetizaciones (o valores promedio de espín), si no que hay configuraciones preferenciales, que nos permiten descartar la solución con $\langle \sigma \rangle = 0$ para $T < T_c$, y descartar la solución negativa para campos magnéticos positivos, y esto lo podemos hacer viendo que la gráfica de la energía libre de Helmholtz, obtenida a partir de (1.27), con respecto a $\langle \sigma \rangle$, y notar que valores minimizan F :

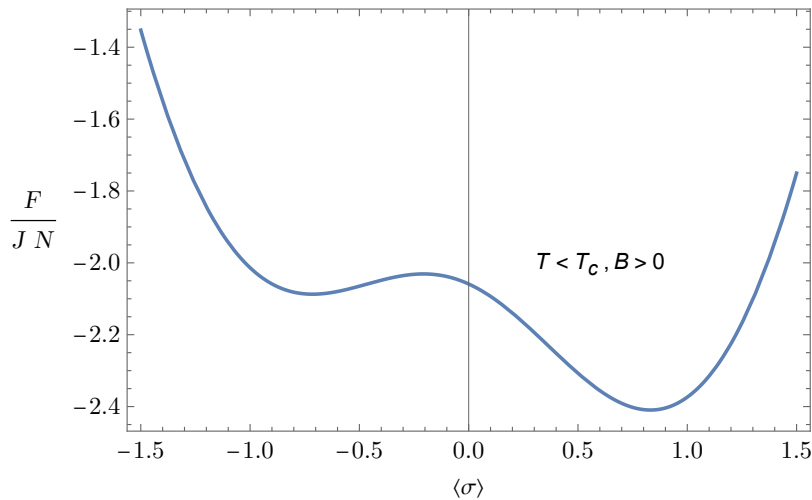
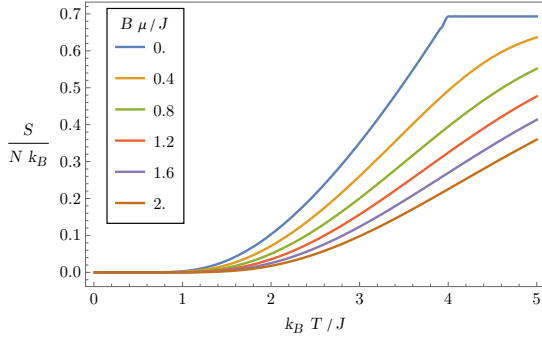
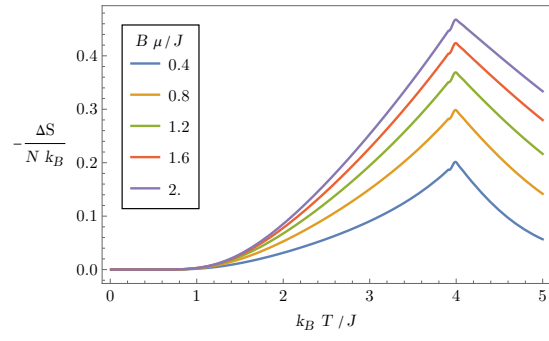


Fig. 1.14: Gráfica de F versus $\langle \sigma \rangle$ para el modelo de Ising de campo medio, con $Q = 4$.

Luego, para poder obtener las cantidades termodinámicas tenemos que obtener numéricamente $\langle \sigma \rangle$ y así saber su dependencia en T y B , con lo que podemos obtener que la gráfica de $S(T)/N$ es la siguiente:



(a) Gráfica de S versus T para el modelo de Ising de campo medio, con $Q = 4$.



(b) Gráfica de ΔS versus T para el modelo de Ising de campo medio, con $Q = 4$.

§3.2.3 Modelo de reloj q-Estados (*q-State Clock*)

En este modelo, trabajamos con una red de espines cuadrada bidimensional de dimensiones $L \times L = N$, y pasamos de tener solamente dos orientaciones posibles para los espines, a q orientaciones posibles (equiprobables) dentro del plano de la red, de tal manera que estas se asemejan a las manillas de un reloj (de ahí su nombre). También, este modelo se ha pensado como una versión discreta del modelo 2D XY, en dónde los espines pueden tomar cualquier ángulo dentro de un círculo en un plano. Varios sistemas se pueden aproximar a este modelo, como la unión Josephson, y láminas delgadas granulares desordenadas superconductoras [16]. Usando el modelo de q-clock, se han llegado a resultados de comportamientos similares entre ambos modelos [17].

Una forma matemática de representar esto, es que el vector de momento magnético local del sitio i puede ser representado como:

$$\mathbf{S}_i = \left(\cos \left(\frac{2\pi}{q} k \right), \sin \left(\frac{2\pi}{q} k \right) \right), \quad k = 0, 1, \dots, q-1 \quad (1.66)$$

De esta manera, el Hamiltoniano del sistema puede ser representado como:

$$H_q = -J \sum_{\langle i,j \rangle} \mathbf{S}_i \cdot \mathbf{S}_j - \mu \mathbf{B} \cdot \sum_i \mathbf{S}_i \quad (1.67)$$

En este Hamiltoniano, vamos a trabajar con J como una unidad de energía, y μB también estará en unidades de energía, con lo que B estará medido en unidades de J/μ , la temperatura en unidades de J/k_B , el calor específico y entropía en unidades de k_B , lo que significa dejar $k_B = J = \mu = 1$.

Sólo trabajaremos con los valores de q pares, ya que nuestro sistema a campo cero presenta una simetría con respecto a la inversión de espín $\mathbf{S}_i \rightarrow -\mathbf{S}_i$, mientras que los q impares rompen esta simetría.

De este modelo ya hay algunos resultados previamente reportados [18], por lo que en este informe, se informará de resultados obtenidos por medio de técnicas computacionales, para así lograr ver las características termodinámicas del modelo, como también las dos transiciones de fase que presenta este, la transición de un estado ferromagnético a uno caracterizado por la formación de vórtices, siendo esta transición llamada Berezinskii–Kosterlitz–Thouless (BKT) [19], y la otra transición es de este último estado, a uno paramagnético [20]. Por estas transiciones, se vuelve interesante ver el efecto magnetocalórico en este modelo, y así también podremos ver cual de estas dos transiciones presenta un mayor MCE [18].

Una forma de trabajar con el modelo, es escribir el Hamiltoniano y su función de partición de forma explícita, o sea, evaluar cada Hamiltoniano para cada configuración, obtener la función de

partición considerando todas las configuraciones, y calcular las cantidades termodinámicas. Nos podemos dar cuenta que en general, vamos a tener q^N configuraciones distintas, lo que para un $q = 2$ y $L = 3$, significa $2^9 = 512$ configuraciones posibles, notando que este número crece exponencialmente, y si bien, podemos usar cálculos computacionales para poder intentar hacer este trabajo posible, en algún momento los tiempos de cálculo se vuelven muy grandes, con lo que nos vemos en necesidad de ocupar métodos numéricos para poder analizar de esta forma el modelo.

§4 Simulación Monte-Carlo

Una manera muy utilizada para abordar el problema de los tiempos de cálculo es el de procedimientos estocásticos para poder calcular integrales, o en este caso directamente, la función de partición. Uno de ellos es el la simulación Monte Carlo, introducido por Metropoli en 1949 [21].

En la simulación Monte Carlo, seguimos la "dependencia temporal" de un modelo en el cuál sus cambios no siguen un camino predeterminado, si no de manera estocástica que depende de números aleatorios generados en la simulación. Con una segunda secuencia de números aleatorios diferentes, los resultados no serán idénticos, pero se acercarán a los iniciales con algún error estadístico [22].

El objetivo de esto es calcular valores de expectación de cantidades observables, y en este trabajo, en especial de la energía interna, que junto a (1.31), encontramos el calor específico C_H , y finalmente, con la (1.19) dejando T_1 cercano a 0, obtenemos la entropía $\Delta(S)_H(T, B)$, y finalmente, haciendo distintas simulaciones con distintos campos, podemos calcular $\Delta S_{iso} = S(T, B) - S(T, 0)$.

Para obtener ΔS_{iso} de $\Delta(S)_H(T, B)$ seguimos el siguiente razonamiento: A temperatura $T = 0$, y campo magnético $B = 0$, nuestro sistema tiende a alinear todos los espines entre sí, con q orientaciones posibles, con lo que la entropía será $S(0, 0) = \ln q$, mientras que a campo magnético $B \neq 0$, los espines además de alinearse, también lo debe hacer con este campo externo, con lo que $S(0, 0) = 0$, con lo que podemos establecer que $S(T, 0) = \ln q + \Delta(S)_H(T, 0)$, y $S(T, B) = \Delta(S)_H(T, B)$, con lo que es $\Delta S_{iso} = \Delta(S)_H(T, B) - (\ln q + \Delta(S)_H(T, 0))$.

§4.1 Estimadores

En general, sabemos que para obtener el valor de expectación de un observable Q hay que ponderar cada valor Q_μ con sus respectivas probabilidades p_μ , y usando el ensemble canónico, tenemos que esto es [23]:

$$\langle Q \rangle = \sum_{\mu} Q_{\mu} p_{\mu} = \frac{\sum_{\mu} Q_{\mu} e^{-\beta E_{\mu}}}{\sum_{\mu} e^{-\beta E_{\mu}}} \quad (1.68)$$

Pero a la hora de realizar cálculos exactos con esta fórmula, esta cantidad se vuelve no práctica de calcular al aumentar el número de partículas disponibles en nuestro sistema. Ahora, si queremos encontrar un estimador de este valor de expectación dado M estados, tendremos que el mejor de ellos es:

$$Q_M = \frac{\sum_i^M Q_{\mu i} p_{\mu i}^{-1} e^{-\beta E_{\mu i}}}{\sum_j^M p_{\mu j}^{-1} e^{-\beta E_{\mu j}}}. \quad (1.69)$$

Nos podríamos preguntar, ¿Cómo podemos elegir estos M estados de tal manera que Q_M sea un estimador adecuado para $\langle Q \rangle$? Si eligiéramos que todas las probabilidades sean iguales, tendríamos:

$$Q_M = \frac{\sum_i^M Q_{\mu i} e^{-\beta E_{\mu i}}}{\sum_j e^{-\beta E_{\mu j}}} \quad (1.70)$$

Pero, resulta que esta decisión es muy mala. Esto dado que solo podemos muestrear un pequeño número de estados en comparación al total. Para dar un ejemplo, consideremos un sistema cúbico Ising de $10 \times 10 \times 10$ espines. Este sistema tendría $2^{1000} \approx 10^{300}$ estados, y un cálculo computacional podría intentar obtener alrededor de 10^8 de esos estados en un buen número de horas en un computador de alta gama, lo que significaría que estaríamos muestreando solamente 1 en 10^{292} estados, y podría ocurrir que el mayor efecto esté dado por un número pequeño de estados, mientras que los otros estados contribuyen muy poco a la simulación. Este efecto es muy importante a la hora de ver temperaturas bajas, en dónde en general los sistemas son dominados por un número pequeño de estados de baja energía dado que no hay suficiente energía térmica.

Entonces, el problema ahora se vuelve en como encontrar estados representativos del sistema, y usar estos para reemplazarlos en (1.69) sin tener que usar muchos términos. La técnica para poder realizar esta selección de estados más importantes entre todos los posibles se llama "Muestreo por importancia" (Importance sampling).

§4.2 Muestreo por importancia

Una situación real que nos puede ayudar a entender la próxima elección de probabilidades, es ver que ocurre con las mediciones de un gas. Consideremos un litro de gas en condiciones ambientales estándar. Este sistema tiene una cantidad de moléculas en el orden de 10^{22} . Velocidades típicas de estas son de alrededor de los 100 m/s, lo que les da una longitud de onda de Broglie de alrededor de los 10^{-10} m. Entonces, cada molécula tendrá 10^{27} estados cuánticos posibles, y el gas completo $(10^{27})^{10^{22}}$ estados posibles. Estas moléculas cambian de un estado a otro al colisionar entre ellas o con las paredes del recipiente, con 10^9 colisiones por segundo, o 10^{31} cambios de estado por segundo. A este ritmo, debería tomar alrededor $10^{10^{23}}$ veces la vida del universo para que nuestro litro de gas pase por todos los estados posibles, pero claramente no necesitamos todo este tiempo para que al medir cualquier cantidad de este gas, obtengamos un valor promedio adecuado. Esto es ya que el gas solo pasa un pequeño número de estados significativos para tener esas cantidades promedio. En efectos prácticos, estos sistemas reales realizan un cálculo Monte Carlo por sí solos [23].

Nosotros podríamos intentar replicar este efecto, y recordando que estos gases se distribuyen de forma Boltzmaniana, podríamos elegir esta distribución como la representativa para nuestro estimador :

$$p_\mu = \frac{e^{-\beta E_\mu}}{Z} \quad (1.71)$$

Con lo que nuestro estimador es:

$$Q_M = \frac{1}{M} \sum_{i=1}^M Q_{\mu i} \quad (1.72)$$

Esta definición de Q_M es mucho mejor que la anterior, especialmente cuando el sistema pasa la mayor parte de su tiempo en un número pequeño de estados en comparación al total.

Ahora queda el como hacer que la selección de estados siga esta distribución de Boltzmann, lo que se logra de forma estándar con los "procesos de Markov".

§4.3 Procesos de Markov

Para nuestros propósitos, un proceso de Markov es un mecanismo en el que, dado un sistema en el estado μ , se genera un nuevo estado ν de forma aleatoria [23]. La probabilidad de general estado ν dado μ es llamada la "probabilidad de transición" $P(\mu \rightarrow \nu)$, y debe cumplir las siguientes condiciones:

- No deben cambiar en el tiempo.
- Deben depender en las propiedades de los estados μ y ν , y no en alguno de los otros estados por los que el sistema ha pasado.
- Debe satisfacer la restricción:

$$\sum_{\nu} P(\mu \rightarrow \nu) = 1, \quad (1.73)$$

dado que se debe generar algún estado cuando trabajamos un estado μ . También, cabe recalcar que se incluye la probabilidad de que se mantenga el mismo estado, o sea, $P(\mu \rightarrow \mu)$.

En las simulaciones Monte Carlo, los procesos de Markov se usan para generar nuevos estados, tal que dado un tiempo suficiente, se produce una sucesión de estados que siguen la distribución de Boltzmann, y así termalizamos nuestro sistema. Para lograr esto, se imponen dos criterios más: ergodicidad y balance detallado.

La condición de ergodicidad para nuestra cadena de procesos de Markov constituye en que, partiendo de un estado, sea posible llegar a cualquier otro estado dado un tiempo suficientemente largo. Este principio se puede ajustar acorde a los recursos disponibles, y el tipo de Monte Carlo a implementar, pero tenemos que tener en cuenta de que al tener menos estados a elegir, nuestra simulación tenga menor precisión.

La condición de balance detallado nos garantiza que si nuestro sistema se termaliza o llega al equilibrio, generamos una distribución de probabilidades de Boltzmann. Para demostrar esto, primero, nos valemos de la llamada ecuación maestra, que nos dice la evolución temporal de las probabilidades de que un sistema se encuentre en un estado μ , es acorde a las tasas de transición de que el sistema llegue al estado μ , y salga del mismo estado:

$$\frac{dp_{\mu}(t)}{dt} = \sum_{\nu} p_{\nu}(t)P(\nu \rightarrow \mu) - \sum_{\nu} p_{\mu}(t)P(\mu \rightarrow \nu) \quad (1.74)$$

Luego, notamos que en el equilibrio, esperamos que la derivada temporal de p_{μ} sea 0, con lo que:

$$\begin{aligned} \sum_{\nu} p_{\mu}P(\mu \rightarrow \nu) &= \sum_{\nu} p_{\nu}P(\nu \rightarrow \mu) \\ p_{\mu} \sum_{\nu} P(\mu \rightarrow \nu) &= \sum_{\nu} p_{\nu}P(\nu \rightarrow \mu) \\ p_{\mu} &= \sum_{\nu} p_{\nu}P(\nu \rightarrow \mu), \end{aligned} \quad (1.75)$$

dónde ocupamos (1.73).

Podríamos pensar que con esto, estaríamos con todas las condiciones necesarias para asegurar la selección adecuada de estados, pero todavía existen problemas con ello, y se puede demostrar este punto.

Partiendo con que la probabilidad de tener el estado ν en un tiempo $t + 1$ viene dada por:

$$p_{\nu}(t + 1) = \sum_{\mu} P(\mu \rightarrow \nu)p_{\mu}(t) \quad (1.76)$$

En notación matricial es:

$$\mathbf{p}(t + 1) = \mathbf{P} \cdot \mathbf{p}(t) \quad (1.77)$$

Con $\mathbf{p}$ el vector con elementos p_μ . Si el proceso alcanza un estado de equilibrio en $t \rightarrow \infty$, tendríamos:

$$\mathbf{p}(\infty) = \mathbf{P} \cdot \mathbf{p}(\infty) \quad (1.78)$$

Pero, es posible que el proceso llegue a un equilibrio dinámico, en dónde $\mathbf{p}$ rota alrededor de distintos valores. Esta rotación es llamada ciclo límite, y se satisface que:

$$\mathbf{p}(\infty) = \mathbf{P}^n \cdot \mathbf{p}(\infty), \quad (1.79)$$

con n la longitud del ciclo. Con esto, la ecuación (1.75) no garantiza que se llegue a una distribución del tipo Boltzmann, y para arreglar este problema, se agrega una nueva condición llamada balance detallado:"

$$p_\mu P(\mu \rightarrow \nu) = p_\nu P(\nu \rightarrow \mu) \quad (1.80)$$

Así, el sistema tiende a transitar en estados $\mu \rightarrow \nu$ con la misma frecuencia que $\nu \rightarrow \mu$. También, se puede demostrar que esta condición elimina los ciclos límite, y una explicación de esto es que, en los ciclos límite, las probabilidades de transición son tales que el sistema sigue un orden cíclico de estados, y para esto, deben ocurrir más transiciones a un estado en particular, que transiciones hacia afuera de ese estado, pero esta condición de balance detallado nos dice que en general, estas deben ser iguales y así logra eliminar los ciclos límite. Esto, con la condición de que el balance detallado nos da al reemplazar p_μ con (1.71), tenemos que:

$$\frac{P(\mu \rightarrow \nu)}{P(\nu \rightarrow \mu)} = \frac{p_\nu}{p_\mu} = e^{-\beta(E_\nu - E_\mu)} \quad (1.81)$$

Y finalmente, con esto logramos garantizar que el proceso de Markov genera estados que siguen probabilidades de la distribución de Boltzmann.

§4.4 Tasas de aceptación

Si bien, se ha detallado en la teoría de procesos de Markov, todavía no se ha establecido cuál es el proceso de Markov correcto a usar en nuestro Monte-Carlo. En general, encontrar este es un gran trabajo de por sí. No siempre es necesario encontrar uno que produzca las probabilidades de transición exactas para una probabilidad de Boltzmann, sino que podemos introducir las llamadas tasas de aceptación, que nos ayudan en esta elección.

Notando que la probabilidad de transición $P(\mu \rightarrow \nu)$ siempre satisface el balance detallado, esto nos da flexibilidad en cómo podemos elegir las otras probabilidades de transición, siempre que ajustemos el valor de $P(\mu \rightarrow \nu)$ de tal manera que se satisfaga (1.73) y no sean valores fuera de $[0, 1]$. Ahora, descompondremos $P(\mu \rightarrow \nu)$ en dos partes:

$$P(\mu \rightarrow \nu) = g(\mu \rightarrow \nu)A(\mu \rightarrow \nu) \quad (1.82)$$

La primera de estas ($g(\mu \rightarrow \nu)$) es la probabilidad de selección, en el que dado un estado μ , se genera el estado ν , y $A(\mu \rightarrow \nu)$ es la tasa de aceptación, que nos dice que si estamos en el estado μ , y se genera ν , aceptamos el cambio al estado ν por una fracción de tiempo, y el resto debería estar en μ . Con esto, tenemos libertad de como elegir $g(\mu \rightarrow \nu)$ ya que (1.81) solo fija las razones de los $A(\mu \rightarrow \nu)/A(\nu \rightarrow \mu)$:

$$\frac{P(\mu \rightarrow \nu)}{P(\nu \rightarrow \mu)} = \frac{g(\mu \rightarrow \nu)A(\mu \rightarrow \nu)}{g(\nu \rightarrow \mu)A(\nu \rightarrow \mu)} \quad (1.83)$$

Con esto dicho, nosotros queremos un algoritmo que se mueva muchísimo entre nuestros estados para así lograr obtener una buena muestra de las cantidades a través de ellos, y la solución a esto es hacer que las tasas de aceptación sean lo más cercanas a 1. Ahora, una forma eficiente de hacer esto es que al tener fijas las razones de los $A(\mu \rightarrow \nu)/A(\nu \rightarrow \mu)$, podemos dejar la mayor de estas como 1, y dejar la otra como el valor que cumple la ecuación (1.83). Con esto, nos aseguramos que al menos se van aceptar los cambios de estado en cierta dirección.

Otro detalle importante, es que en la práctica, lo mejor es dejar las características de los estados μ y ν como factores que afectan en su mayor parte solamente a $g(\mu \rightarrow \nu)$, y poco a las tasa de aceptación.

§4.5 Algoritmo Metropolis

El algoritmo a utilizar en el trabajo es el Metropolis, introducido por Nicolas Metropoli y sus compañeros de trabajo en 1953, usado para simular gases de esferas duras [24]. Cabe recalcar que si bien, existe una gran cantidad de algoritmos, este es uno de los más utilizados.

Un hecho importante es que las energías en el equilibrio térmico tienen fluctuaciones muy pequeñas en comparación a la energía del sistema completo, o en otras palabras, el sistema pasa la mayor parte del tiempo en estados en un rango pequeño de energías y no hay muchas transiciones a estados que cambien la energía del sistema de forma dramática. La forma más fácil de usar esto es considerar cambios de estado en los cuáles solo hayan variaciones en un solo espín, o en aglomeraciones de ellos, llamadas "single espín updates", o "Cluster updates", respectivamente.

Ahora, en el algoritmo Metropolis, se eligen las probabilidades de selección $g(\mu \rightarrow \nu)$ de todos los estados ν iguales. Notando que nuestro sistema tendrá N espines, y usando los single espín updates, vamos a tener N posibles estados ν a los que podemos llegar dejando fijo μ . Por lo tanto, tenemos N probabilidades de selección $g(\mu \rightarrow \nu)$ distintas de 0, con el valor:

$$g(\mu \rightarrow \nu) = \frac{1}{N} \quad (1.84)$$

Con esto, la condición de balance detallado (1.83) va a ser:

$$\frac{P(\mu \rightarrow \nu)}{P(\nu \rightarrow \mu)} = \frac{g(\mu \rightarrow \nu)A(\mu \rightarrow \nu)}{g(\nu \rightarrow \mu)A(\nu \rightarrow \mu)} = \frac{A(\mu \rightarrow \nu)}{A(\nu \rightarrow \mu)} = e^{-\beta(E_\nu - E_\mu)} \quad (1.85)$$

Para poder maximizar la tasa de aceptación, la idea es dejar el mayor de estos como 1, y ajustar el otro. En este caso, se hace el siguiente raciocinio: Teniendo los estados μ y ν , si la energía del estado ν es menor que la de μ , el cambio de estado $A(\mu \rightarrow \nu)$ es 1, y en caso contrario, tendríamos que $A(\nu \rightarrow \mu)$ tiene que ser 1, y $A(\mu \rightarrow \nu)$ debe ser $e^{-\beta(E_\nu - E_\mu)}$ para poder mantener la condición anterior. Por lo tanto, tendríamos que $A(\mu \rightarrow \nu)$ es:

$$A(\mu \rightarrow \nu) = \begin{cases} e^{-\beta(E_\nu - E_\mu)} & , \text{ Si } E_\nu > E_\mu \\ 1 & , \text{ Si } E_\nu < E_\mu \end{cases} \quad (1.86)$$

Ahora, queda ver si en caso que $E_\nu > E_\mu$ el cambio realmente ocurre o no (Hasta ahora, solo hemos visto la probabilidad con que ocurra), y esto lo hacemos eligiendo un número r al azar entre 0 y 1, con distribución uniforme, y si $r < e^{-\beta(E_\nu - E_\mu)}$, aceptamos el cambio, y en caso contrario, lo dejamos igual.

De forma resumida, los pasos a seguir en el algoritmo Metropolis, para trabajar en el sistema de q -clock, son [22]:

1. Elegimos un estado inicial, por ejemplo, que todos los espines estén alineados en cierta dirección, o todos apuntando en direcciones aleatorias uno con respecto a otro.

2. Elegimos un sitio i .
3. Calculamos el cambio de energía $\Delta E = E_\nu - E_\mu$ que resultaría si cambiáramos el espín del sitio i , a alguno de los $q - 1$ posibles.
4. Si $\Delta E < 0$, aceptamos el cambio, dejamos este como nuestra nueva configuración inicial, y volvemos al paso 2.
5. Si $\Delta E > 0$, elegimos un número aleatorio r con $0 \leq r < 1$.
6. Si $r < e^{-\beta\Delta E}$, cambiamos el espín, dejamos este como nuestra nueva configuración inicial, y volvemos al paso 2.
7. Si $r > e^{-\beta\Delta E}$, no se acepta el cambio, y volvemos al paso 2.

En general, se mide el "tiempo" de este proceso en tiempo Monte Carlo, con medida el paso Monte Carlo, correspondiente a que nuestro algoritmo recorra cada sitio de la red una vez.

§4.6 Mediciones

En general, nuestra toma de datos ocurre una vez que nos aseguramos que el sistema se encuentra termalizado, pero con ello, nos entra la duda de, ¿Cuanto tiempo tarda el sistema en llegar a la termalización? Podríamos hacer un gráfico de nuestras cantidades con respecto al número de pasos Monte Carlo, ver en que punto llega a un equilibrio o se estabiliza alrededor de un valor, y luego empezar a tomar datos.

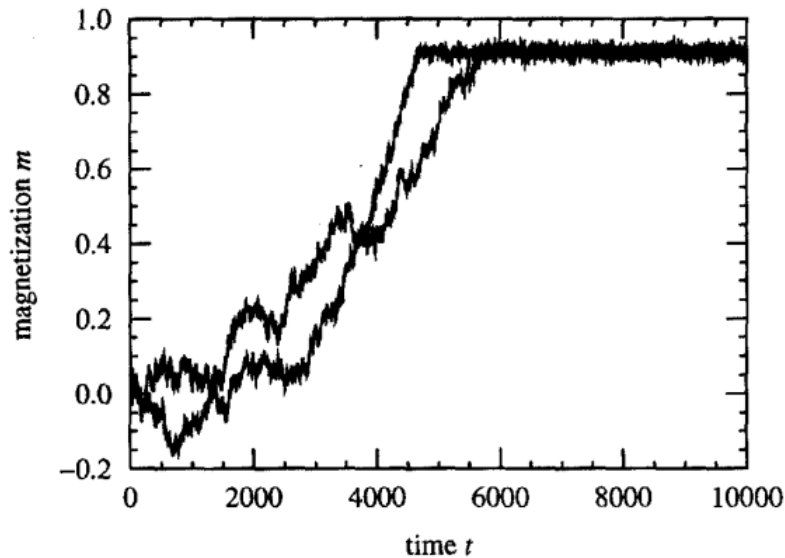


Fig. 1.16: Magnetización de un sistema Ising 2D 100x100 respecto al tiempo (Pasos Monte Carlo), para dos simulaciones distintas con estados iniciales diferentes. Para el tiempo $t = 6000$, ambos fluctúan con respecto a un valor promedio igual, con lo que se concluye que ambos sistemas han termalizado [23].

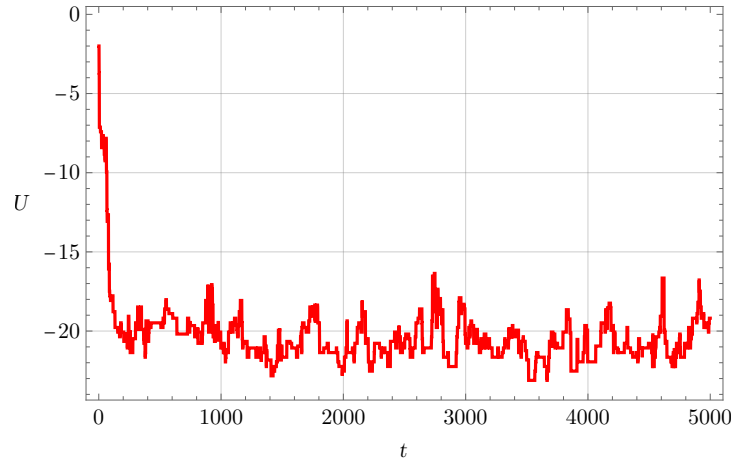


Fig. 1.17: Energía interna de un sistema q-clock 4x4 con $q = 8$, respecto al tiempo (Pasos Monte Carlo) y variables $T = 0.5$, $B = 0$. Notamos que la termalización se alcanza en un pequeño número de pasos.

Para ello, se define una cantidad llamada tiempo de correlación τ , que corresponde al tiempo para que un sistema en un estado A , genere un estado B no correlacionado, de tal forma que A pueda ser usado para los promedios estadísticos. Esto ocurre ya que las cadenas de Markov generan estados poco diferentes uno del siguiente, lo que a la hora de medir, afecta como un sesgo, por lo que se espera un tiempo τ para asegurar que estos dos estados sean independientes. Para encontrar este tiempo, se encuentra la función autocorrelación de la cantidad a medir, definida como:

$$\chi(t) = \int dt' [m(t') - \langle m \rangle] [m(t' + t) - \langle m \rangle] \quad (1.87)$$

Como ejemplo, en el caso de la magnetización en un sistema Ising 2D, podríamos observar la siguiente gráfica.

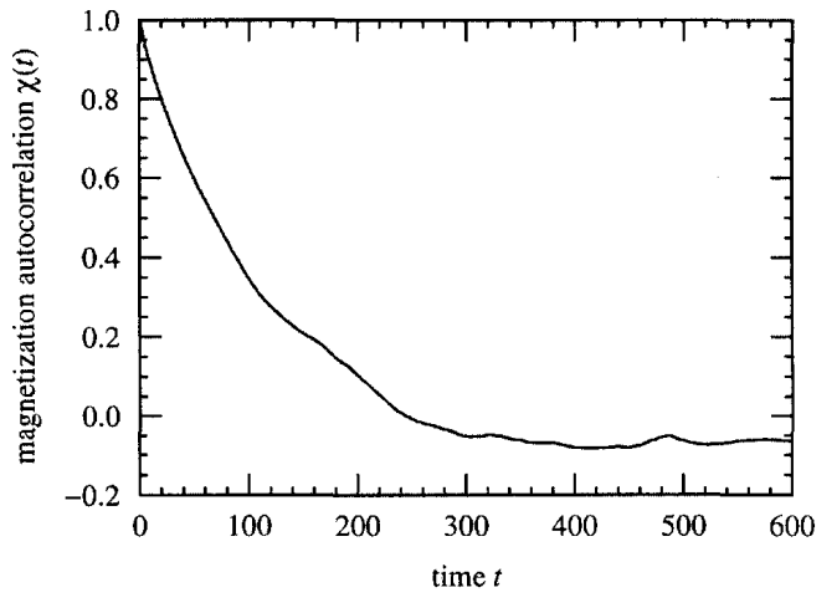


Fig. 1.18: Autocorrelación de la magnetización en un sistema Ising 2D 100x100 respecto al tiempo [23].

Se espera que la autocorrelación sea una función exponencial decauyente, de la forma:

$$\chi(t) \sim e^{-t/\tau} \quad (1.88)$$

Con esto, se espera que se pueda encontrar una configuración independiente cada 2τ , con lo que el número de mediciones totales que se puede hacer, si la simulación durara t_{max} , sería:

$$n = \frac{t_{max}}{2\tau} \quad (1.89)$$

De todas formas, es muy común tomar datos en intervalos menores que el del tiempo de correlación. Por ejemplo, en el caso del algoritmo Metropoli, podríamos tomar una medición cada paso Monte Carlo (vez que consideramos toda la red). Esto ya que no conocemos el tiempo de correlación hasta una vez que nuestra simulación termine, y queremos asegurarnos de aunque sea tomar una medición independiente cada 2 tiempos de correlación, y también necesitamos suficientes datos para poder calcular nuestra función de autocorrelación con precisión.⁴

§4.7 Simulación computacional

Para trabajar en esta tesina, se hizo uso de dos programas. Los cálculos teóricos, y gráficos se obtuvieron con Mathematica, mientras que el algoritmo Monte Carlo se realizó con 10000 pasos Monte Carlo para termalización, y la gran cantidad de pasos nos asegura que todos los sistemas que analicemos estarán termalizados (ver figuras 1.16 y 1.17), y 50000 pasos para el cálculo de promedios. Una vez calculados los últimos, se avanzó en temperaturas con pasos de 0.01, partiendo de $T = 0.01$ a $T = 5.00$, y se repite todo el procedimiento para distinto campo magnético. Esto se llevo a cabo en un código de C++, compilado con g++, del cuál se obtuvo los datos usados.⁵

Cabe destacar, que las redes usadas fueron de un largo $(L + 2) \times (L + 2)$, esto ya que ocupamos las condiciones de borde libre, en dónde dejamos el espín de los bordes como 0, y esta elección proviene de que esta condición es mejor para emular el hecho de trabajar en sistemas pequeños.

⁴El tema de simulación Monte Carlo se trata con mayor profundidad en los libros de D. Landau y K. Binder *A Guide to Monte Carlo Simulations in Statistical Physics*, y M. E. J. Newman y G. T. Berkema *Monte Carlo Methods in Statistical Physics*.

⁵En el anexo §3 puede encontrar más información sobre estos códigos.

Capítulo 2

Resultados y discusión

§1 Resultados analíticos

§1.1 Resultados y discusión para 3x3

Empezamos la discusión viendo los resultados analíticos de la red 3x3, la cuál, por nuestra potencia computacional, sólo la calculamos para $q = 2$ y $q = 4$. Los resultados están en las siguientes figuras:

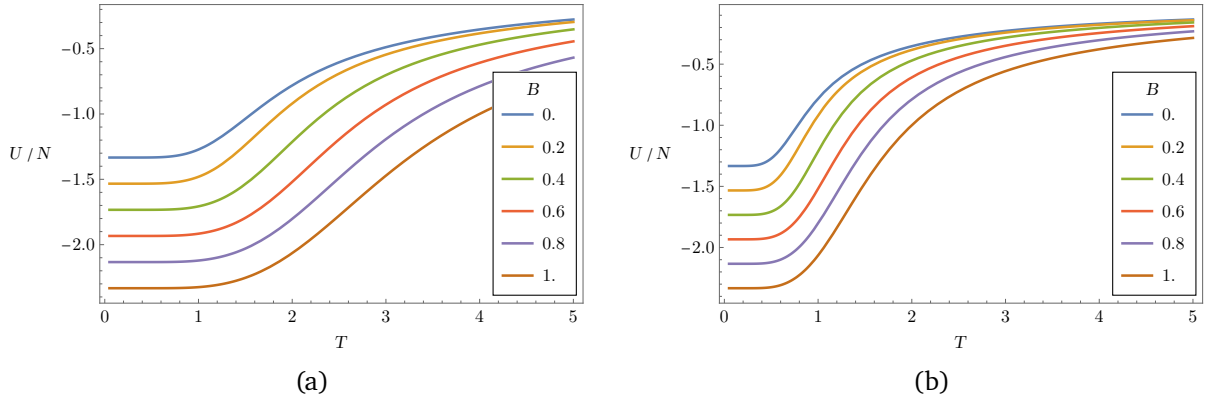


Fig. 2.1: Energía interna U/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 3x3, con $q = 2$ (izquierda) y $q = 4$ (derecha).

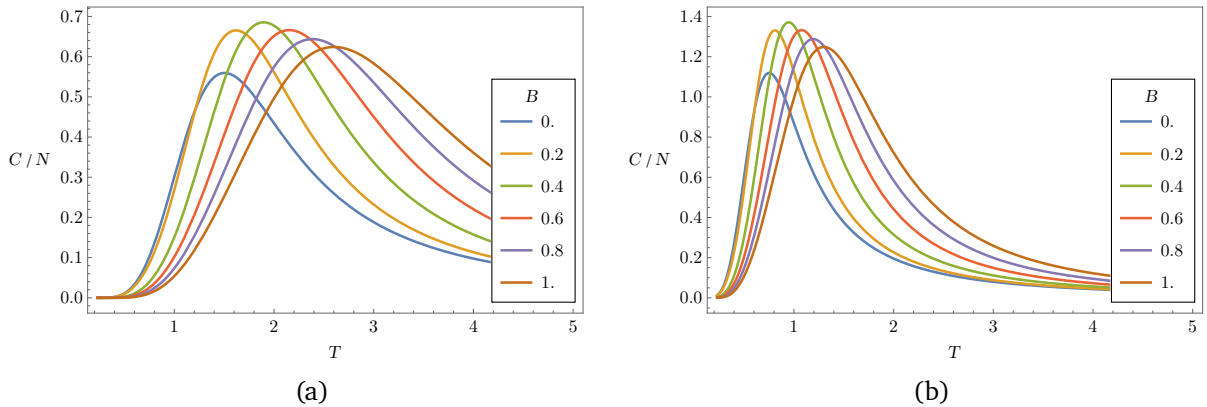


Fig. 2.2: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 3x3, con $q = 2$ (izquierda) y $q = 4$ (derecha).

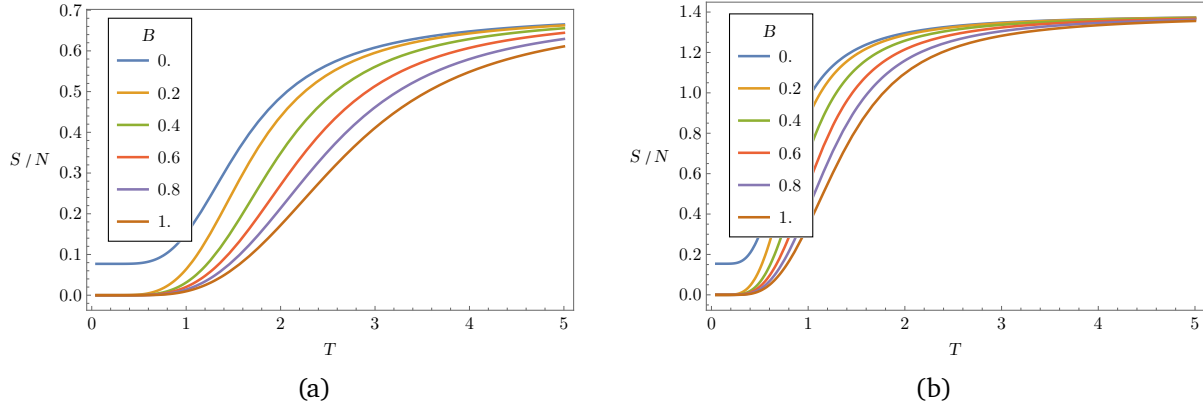


Fig. 2.3: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 3×3 , con $q = 2$ (izquierda) y $q = 4$ (derecha).

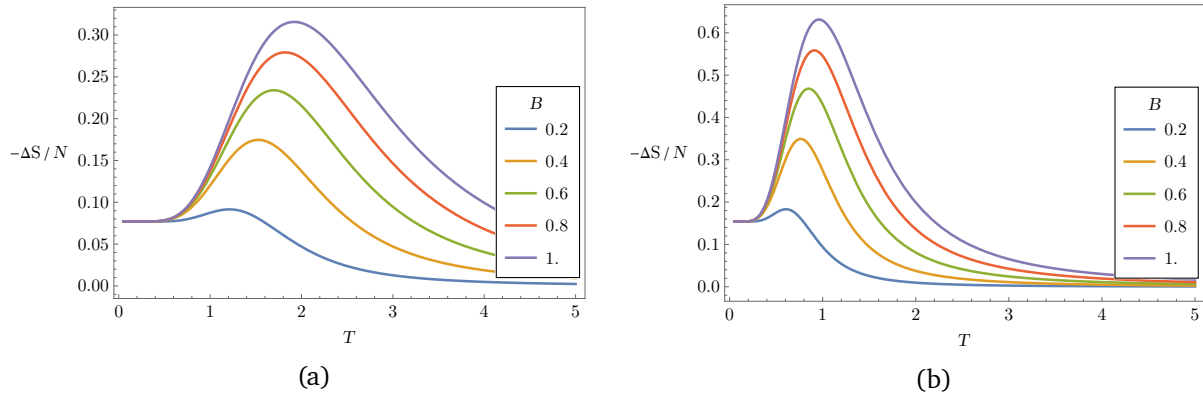


Fig. 2.4: Variación de entropía $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 3×3 , con $q = 2$ (izquierda) y $q = 4$ (derecha). Cabe destacar que no obtuvimos la gráfica para $B = 0$ dado que es la nula ($\Delta S_{iso} = 0$).

Dado que trabajamos con $q < 5$, solo se observa un peak en C (figura 2.3), correspondiente a la transición de un sistema ferromagnético, a uno paramagnético desordenado.

Primero, a temperaturas bajas y campo nulo, vemos que se cumple nuestra predicción de la entropía, ya que esta adquiere el valor de $(\ln q)/N$, mientras que al aplicar campo magnético se rompe la degenerancia de la energía y la entropía adquiere el valor 0. Por otro lado, a temperaturas altas, la interacción de exchange y acople a campo magnético son muy bajas al compararlas con las fluctuaciones térmicas, en la que cada espín es independiente uno del otro, y tenemos q^N configuraciones distintas del sistema, correspondiente a una entropía de $S/N = \ln q$.

El efecto de aplicar campos magnéticos es un corrimiento del peak, lo que se puede deber a que el estos campos hacen favorecer la fase ferromagnética sobre la desordenada, con lo que la temperatura de transición debe aumentar. También, se observa que a mayor campo aplicado, mayor es nuestro efecto magnetocalórico, lo cuál es esperable, ya que debería existir una mayor diferencia en el orden entre los estados sin campo magnético, comparados con los que tienen un campo. Igualmente, este campo hace que la energía base del sistema disminuya.

En relación a q podemos ver las figuras 2.2, 2.3, y 2.4 se observan mayores valores al comparar los resultados de $q = 2$ con los de $q = 4$, lo que se puede deber a que en el último caso, el sistema tiene un mayor número de configuraciones posibles comparado al primero.

§2 Resultados computacionales

§2.1 Resultados

§2.1.1 Red 10x10

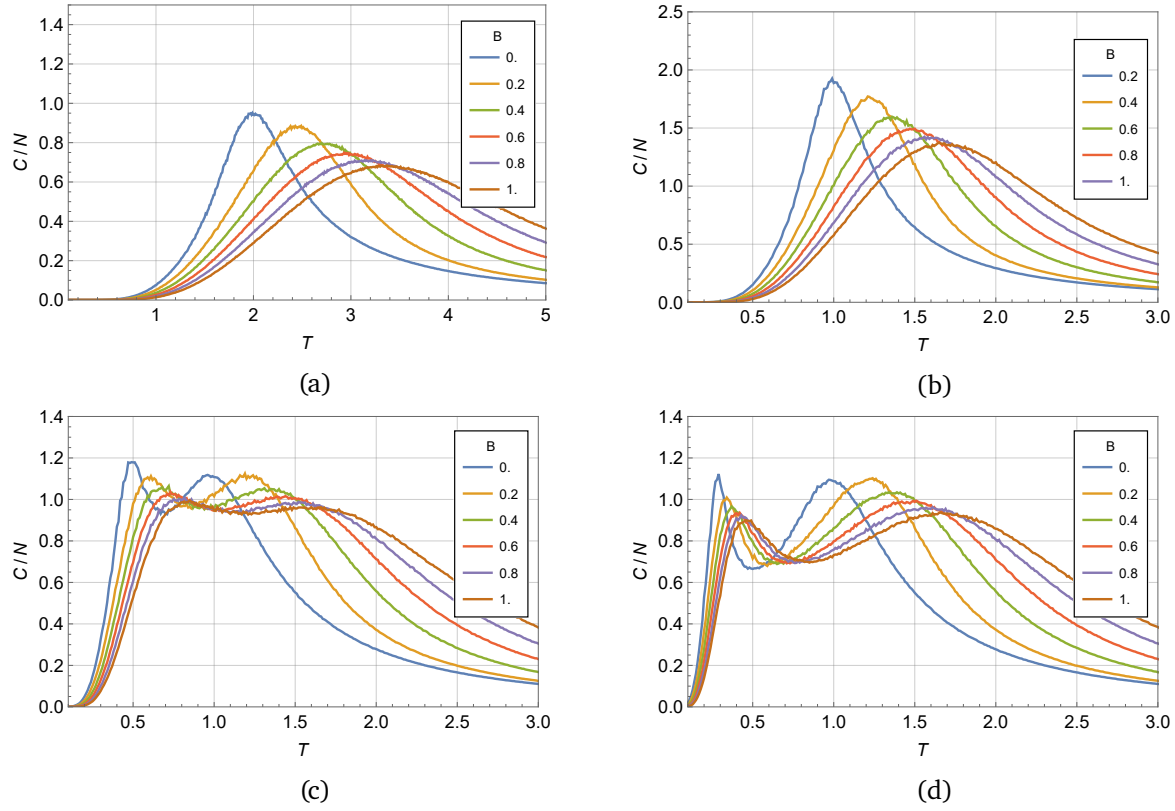


Fig. 2.5: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 10x10, con $q = 2, 4, 6, 8$ respectivamente.

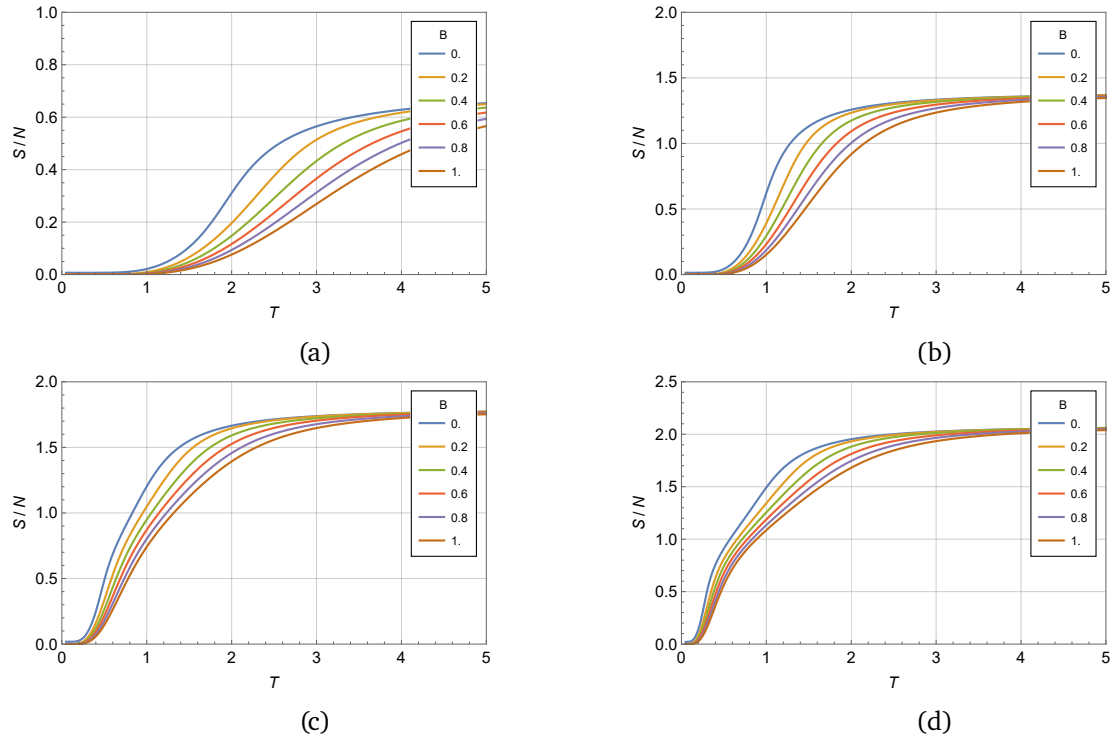


Fig. 2.6: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 10×10 , con $q = 2, 4, 6, 8$ respectivamente.

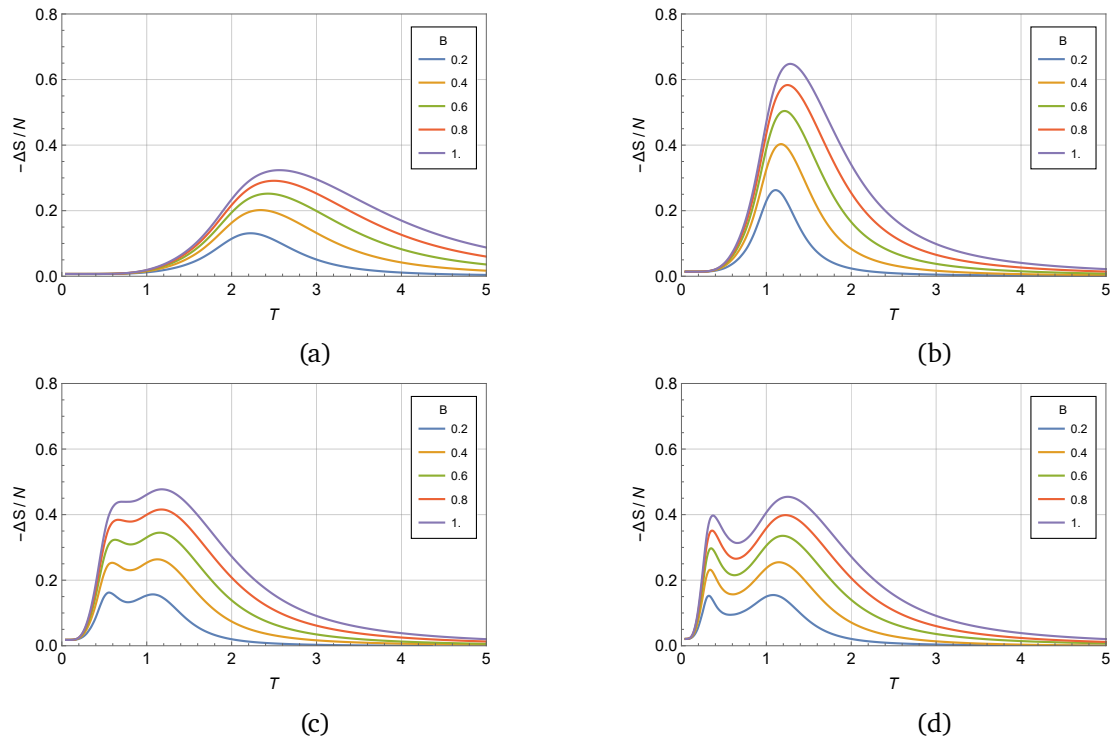


Fig. 2.7: $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 10×10 , con $q = 2, 4, 6, 8$ respectivamente.

§2.1.2 Red 16x16

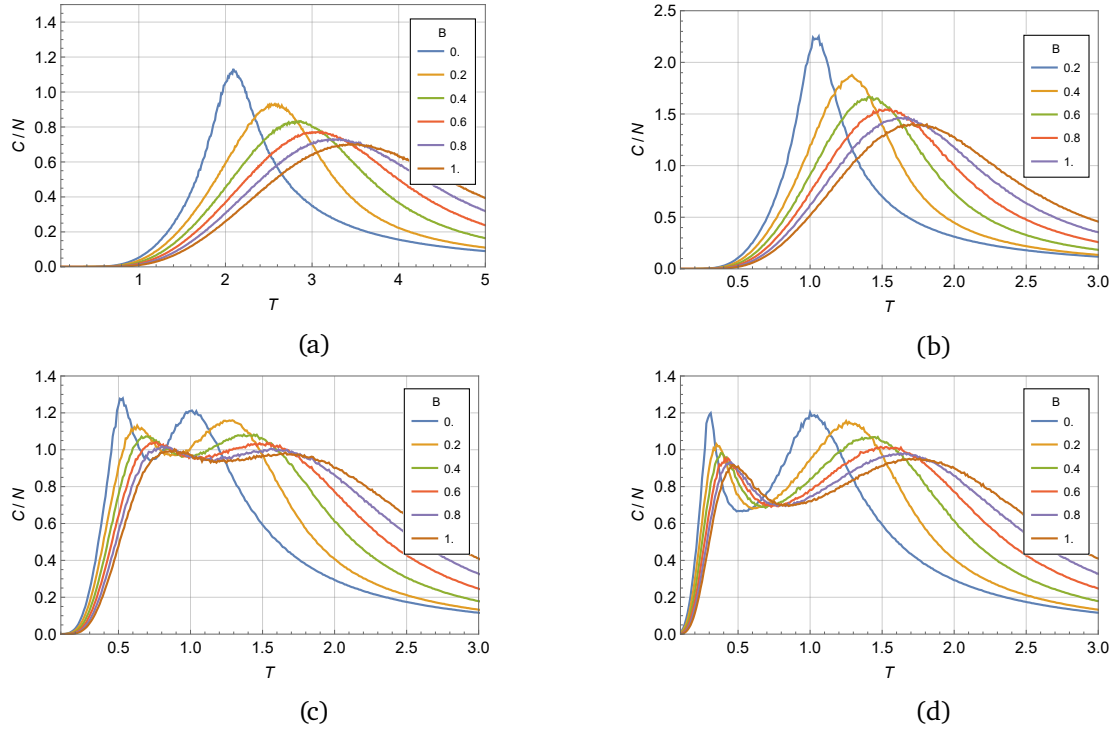


Fig. 2.8: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 16x16, con $q = 2, 4, 6, 8$ respectivamente.

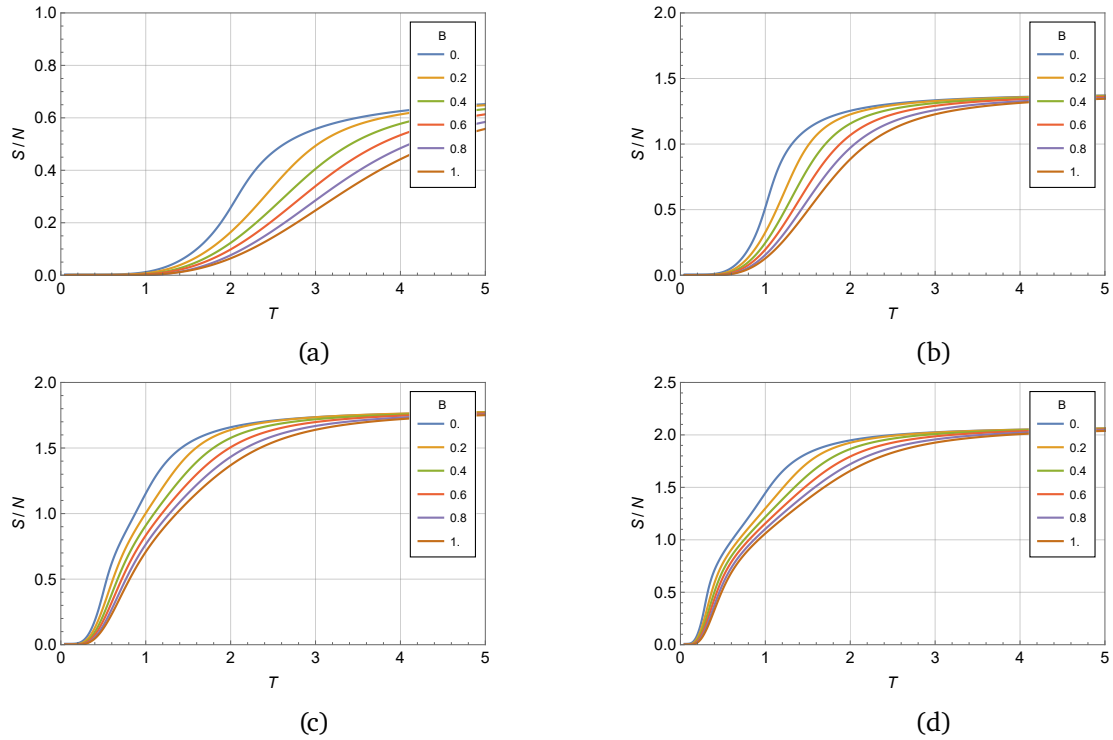


Fig. 2.9: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 16x16, con $q = 2, 4, 6, 8$ respectivamente.

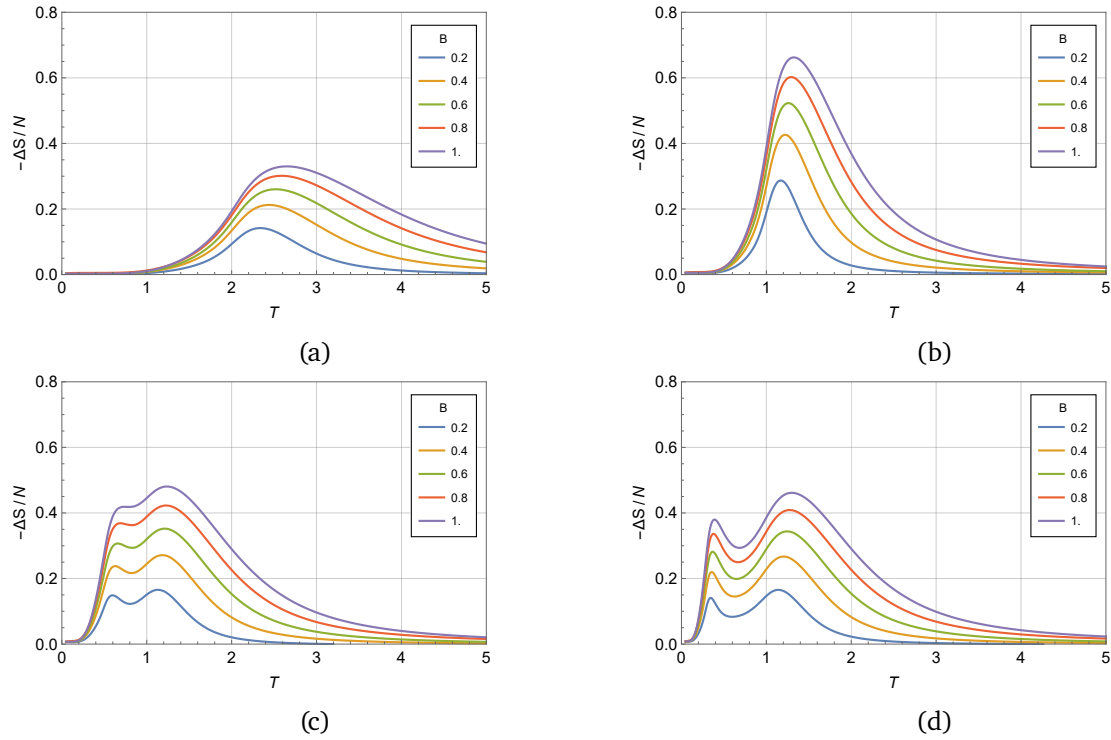


Fig. 2.10: $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 16×16 , con $q = 2, 4, 6, 8$ respectivamente.

§2.1.3 Red 32x32

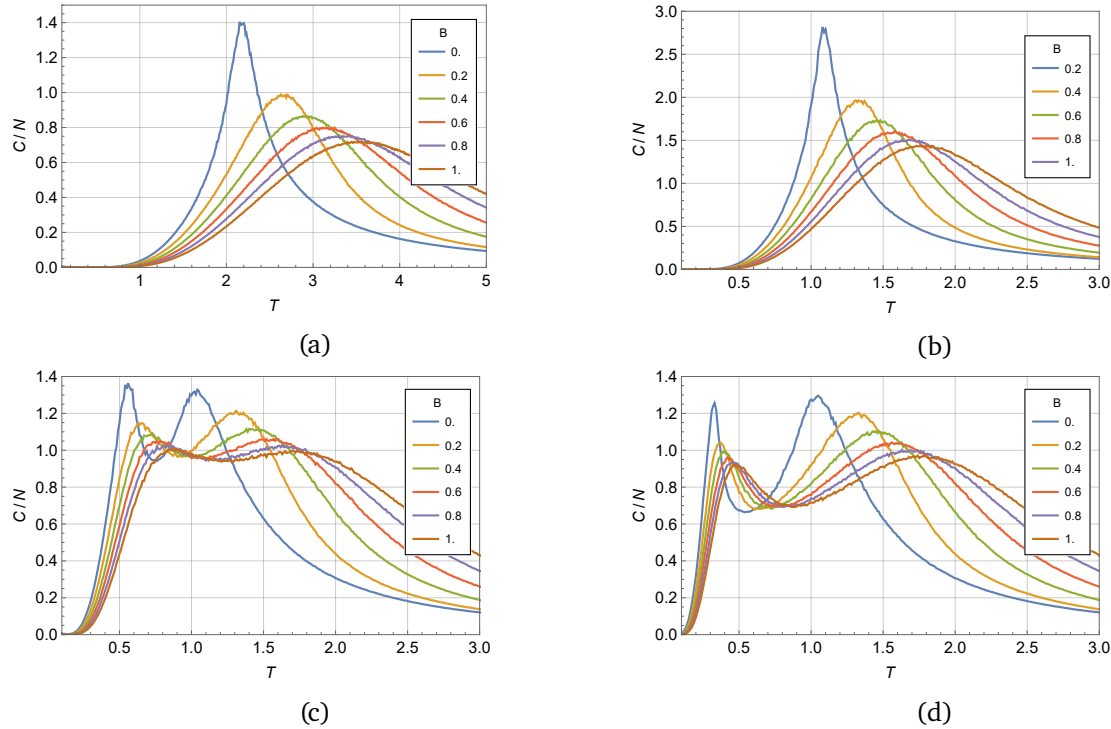


Fig. 2.11: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 32×32 , con $q = 2, 4, 6, 8$ respectivamente.

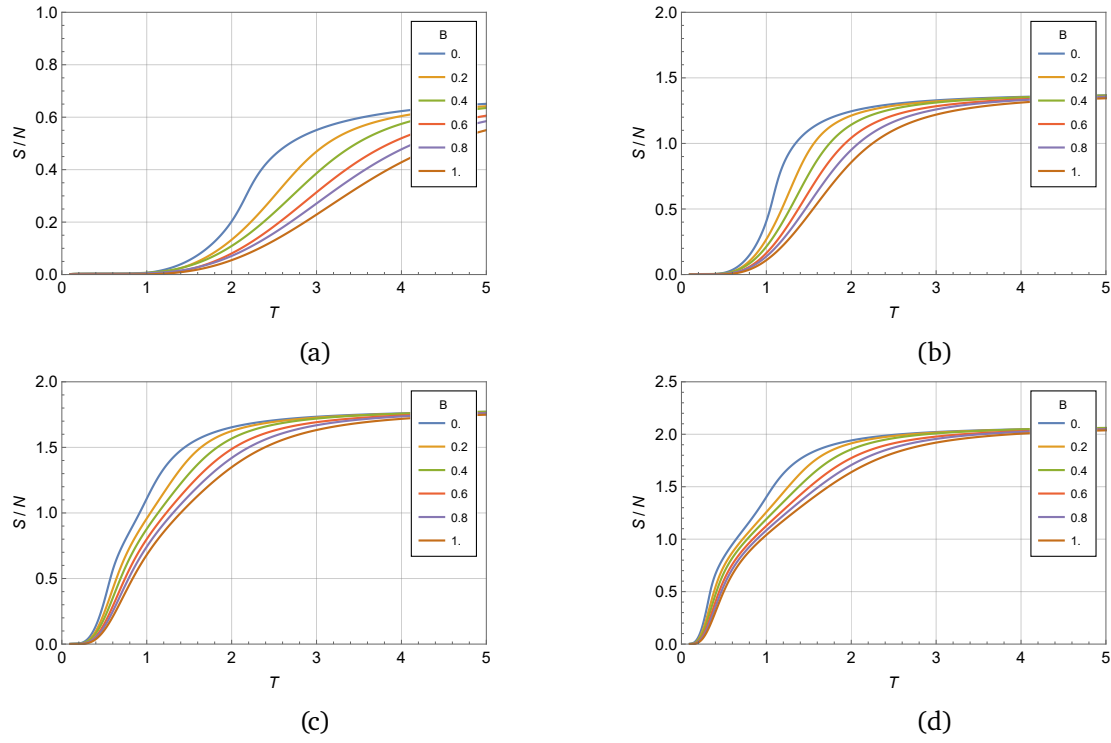


Fig. 2.12: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 32×32 , con $q = 2, 4, 6, 8$ respectivamente.

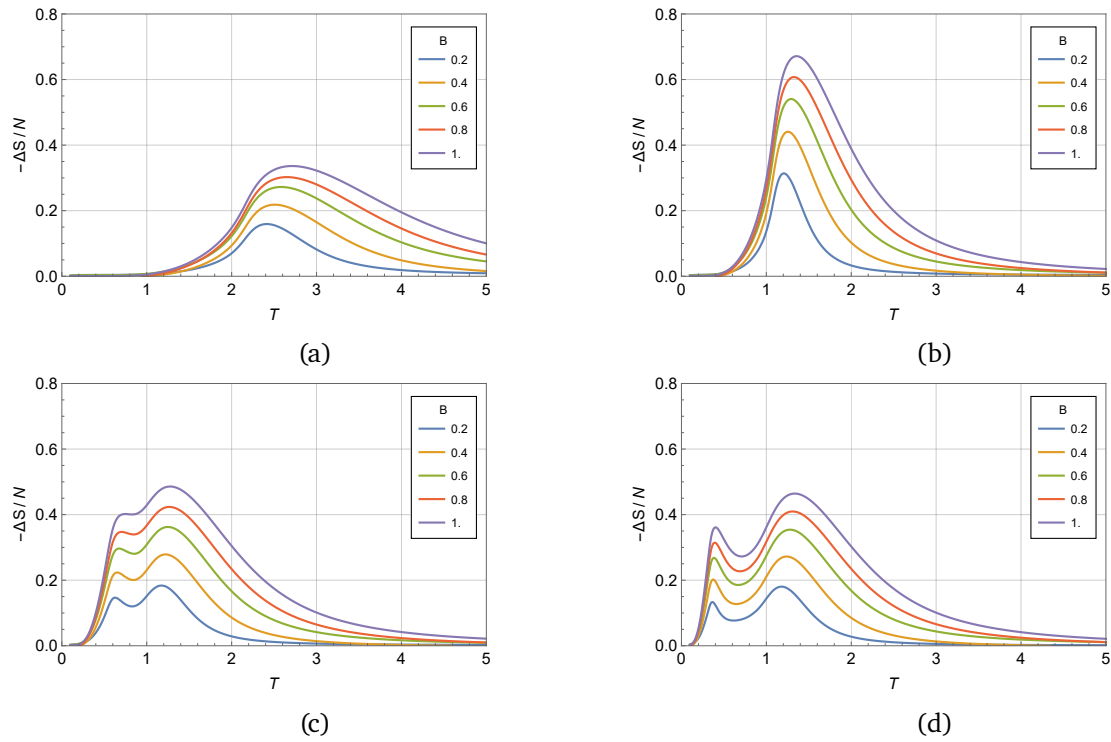


Fig. 2.13: $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 32×32 , con $q = 2, 4, 6, 8$ respectivamente.

§2.1.4 Red 64x64

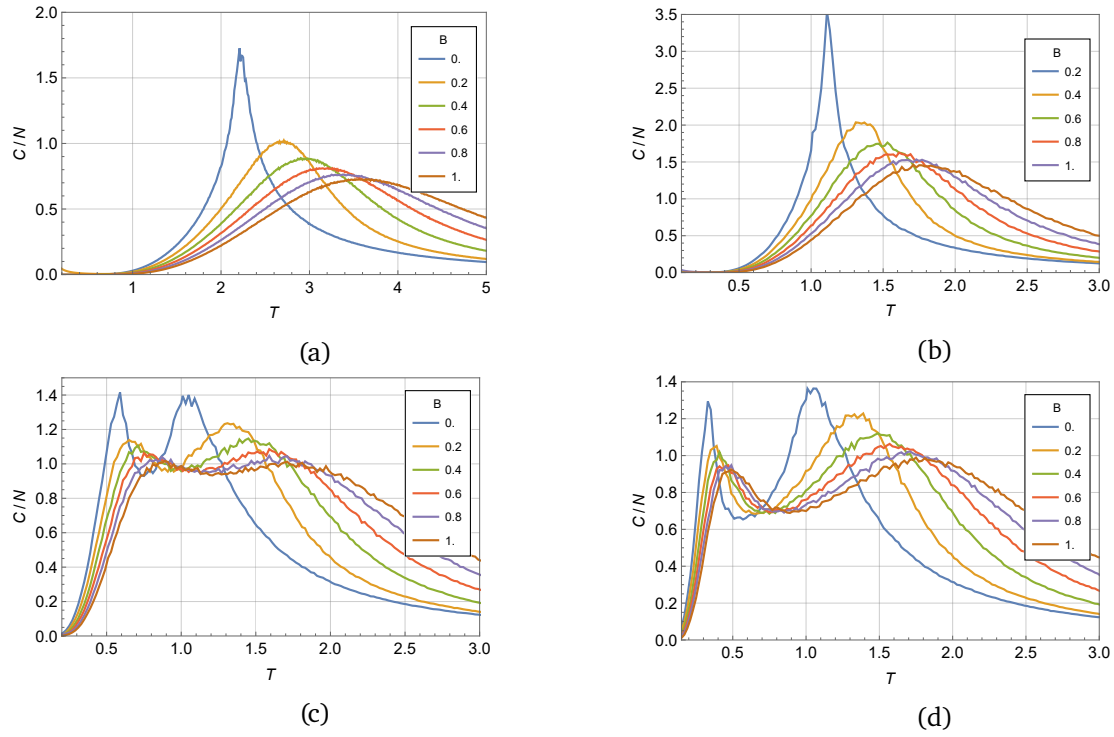


Fig. 2.14: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 64×64 , con $q = 2, 4, 6, 8$ respectivamente.

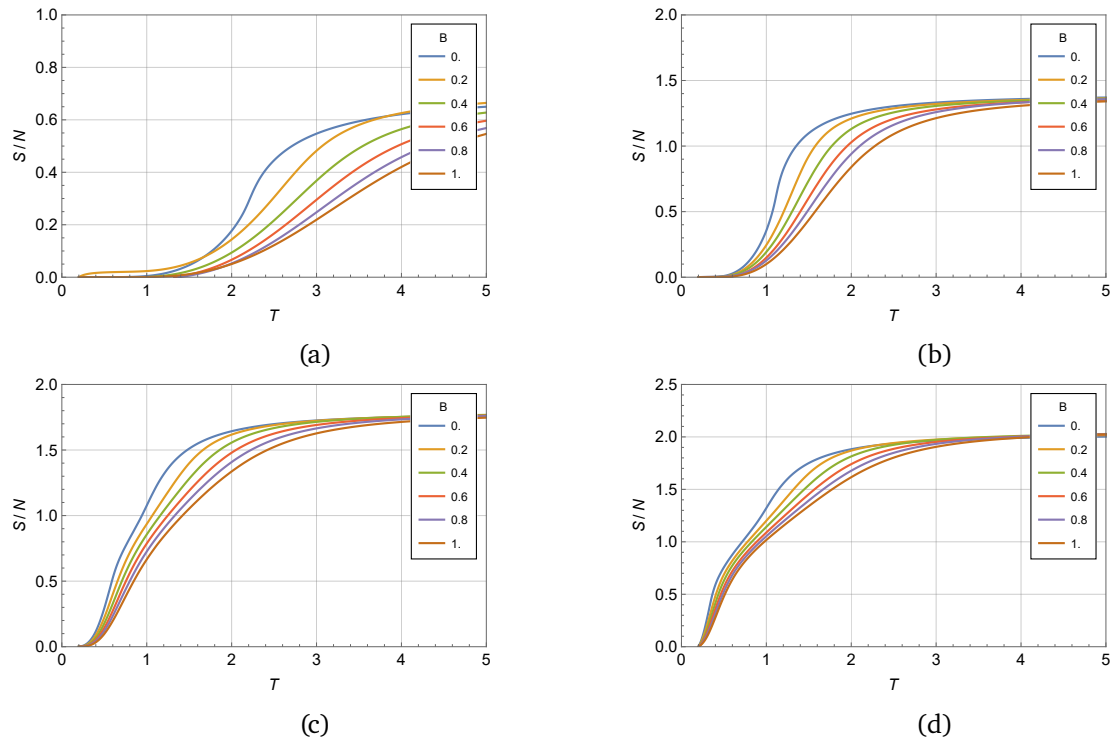


Fig. 2.15: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 64×64 , con $q = 2, 4, 6, 8$ respectivamente.

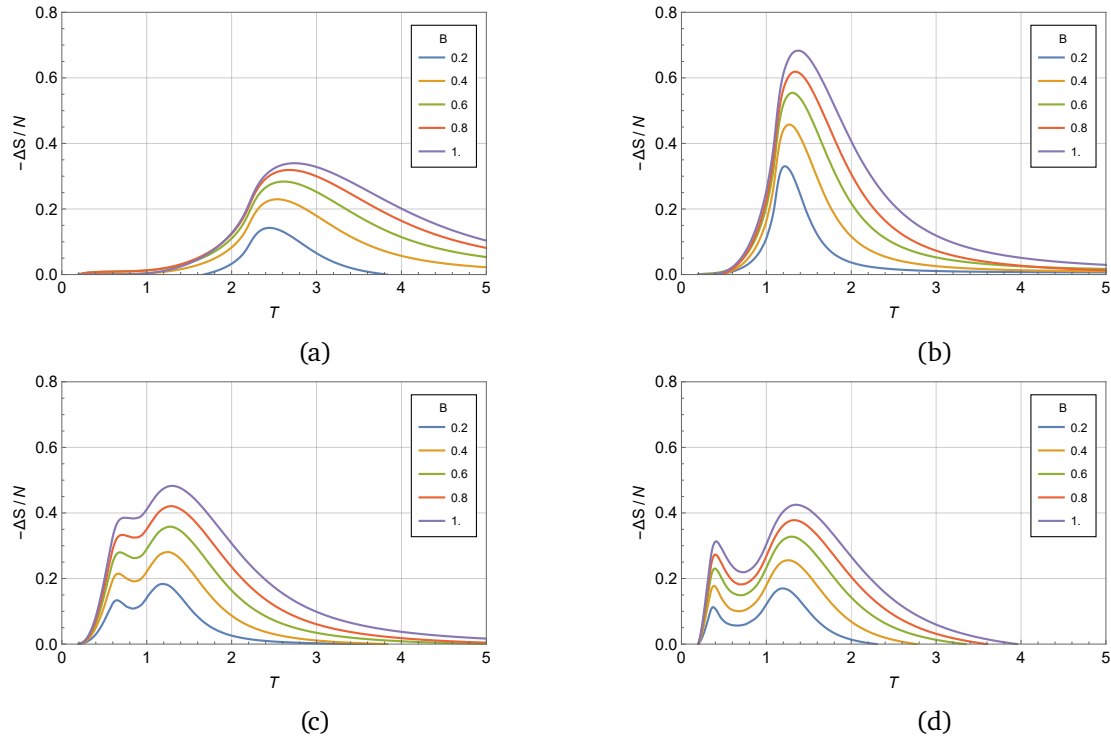


Fig. 2.16: $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 64×64 , con $q = 2, 4, 6, 8$ respectivamente.

§2.1.5 Red 128x128

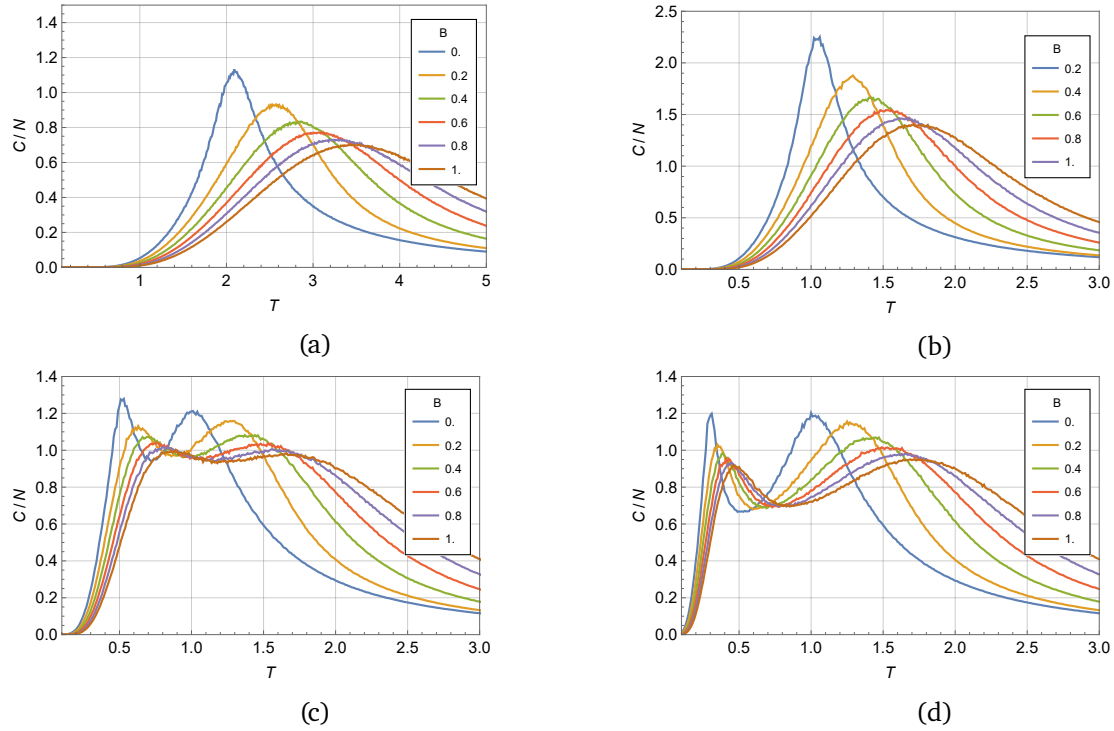


Fig. 2.17: Calor específico C/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 128×128 , con $q = 2, 4, 6, 8$ respectivamente.

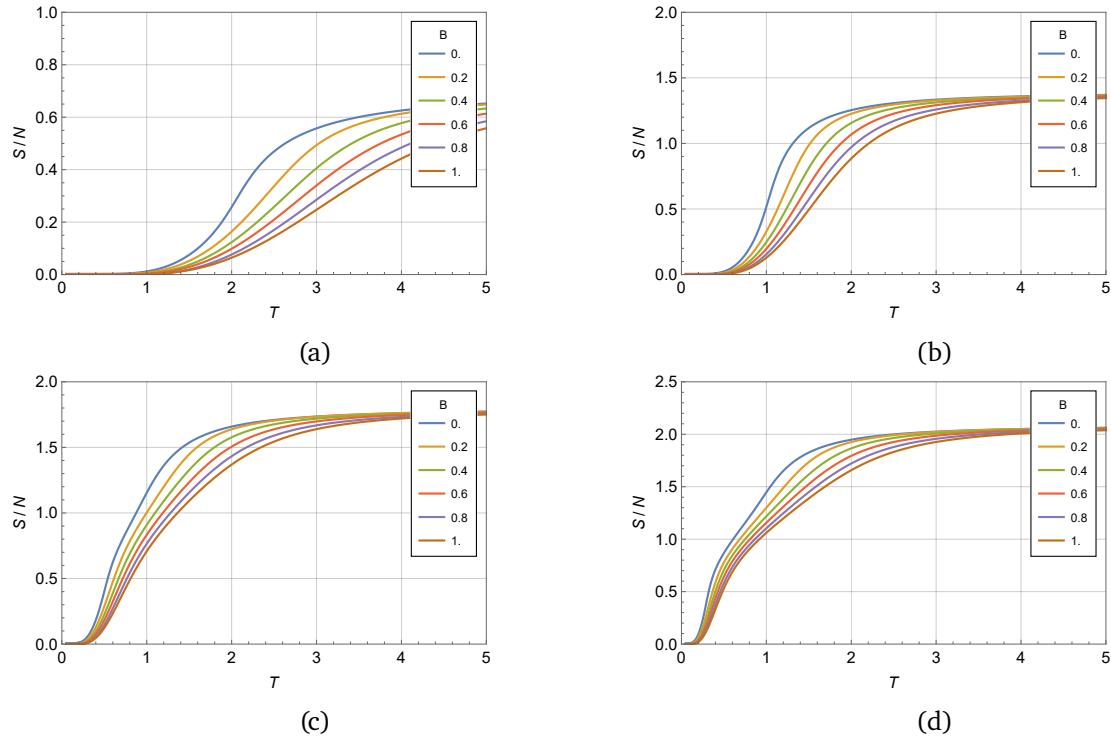


Fig. 2.18: Entropía S/N en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 128×128 , con $q = 2, 4, 6, 8$ respectivamente.

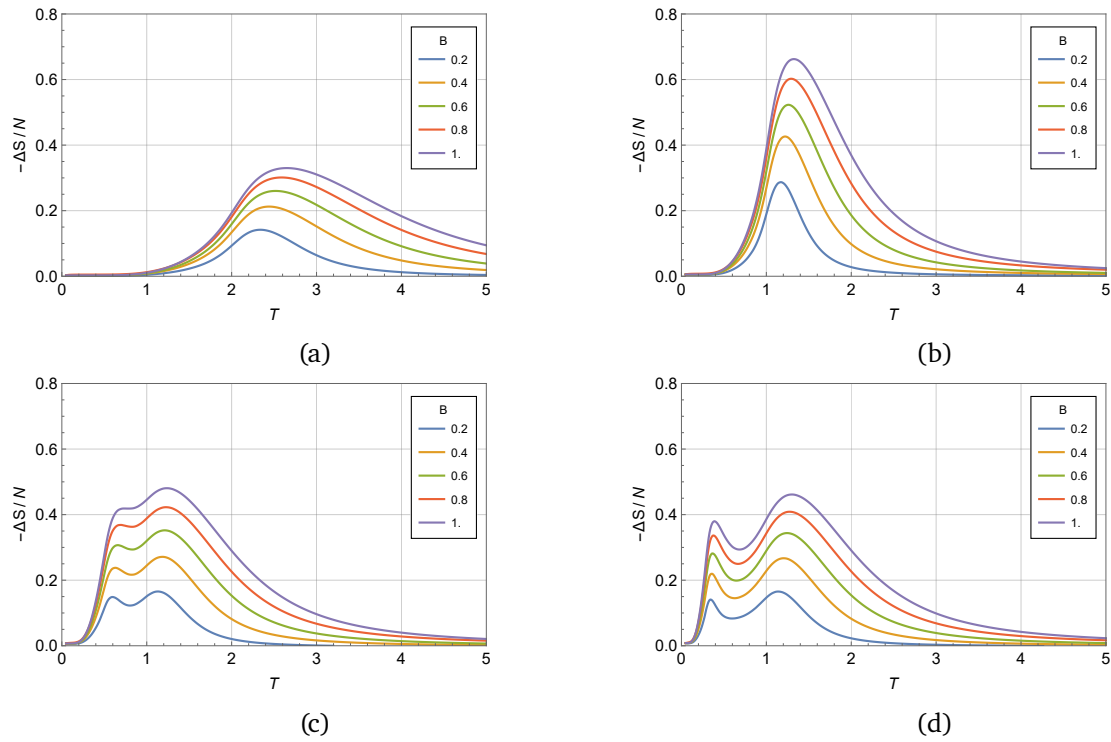


Fig. 2.19: $\Delta S_{iso}/N$ en función de T para distintos campos B apuntando al eje y , para el modelo de q -clock para una red de 128×128 , con $q = 2, 4, 6, 8$ respectivamente.

§2.2 Discusión

Primero, los gráficos del calor específico por espín C/N son los obtenidos con los datos directos de la simulación computacional, mientras que los otros son obtenidos al interpolar y obtener los datos en puntos específicos, para luego interpolar, y dado el gran número de puntos, esperamos que esta sea una buena simulación de la situación.

En las gráficas de C/N (figuras 2.5, 2.8, 2.11, 2.14 y 2.17), observamos para $q = 2$ y 4 una única transición de fase igual a los casos estudiados anteriormente, siendo esta transición correspondiente a un estado ferromagnético a paramagnético desordenado, mientras que para $q = 6$ y 8 observamos otra transición más, en dónde entre los dos estados nombrados anteriormente, vemos una fase BKT, en la cual existen estados de vórtices entre los espines, y la transición de fase a este estado ocurre a temperaturas bajas, disminuyendo esta temperatura de transición al aumentar q . También, en general se observa que el mayor peak en C/N es con $q = 4$, y este aumenta a medida que aumentamos el tamaño de la red. Que el mayor peak ocurra con $q = 4$ puede deberse a que es el menor estado con solo una única transición, y que la presencia de dos transiciones hace que el cambio de estados sea más gradual, disminuyendo la capacidad calorífica de cada peak.

También, se observa que el calor específico disminuye su peak al aumentar el campo magnético, lo cual puede deberse a una transición de fase más gradual por el hecho de que el campo se opone al "desorden" de los espines. Notamos que a temperaturas altas, el calor específico decae a 0.

En relación a la entropía por espín S/N (figuras 2.6, 2.9, 2.12, 2.15 y 2.18), se logra apreciar un número de inflexiones en ella correspondiente a cuantas transiciones de fase presenta el sistema con un q respectivo, y el valor de $S(T = 0, B = 0) = \ln q$ no se logra observar para L grandes, ya que dividimos por $N = L^2$ para obtener las gráficas. También, se observa que se cumple lo predicho en relación al valor de S/N con T altas, siendo teniendo de asíntota $\ln q$, independiente de B , sólo que a B mayores se necesita más temperatura para acercarse a esta asíntota.

Finalmente, en relación a $\Delta S_{iso}/N$ (figuras 2.7, 2.10, 2.13, 2.16 y 2.19), observamos peaks en las mismas temperaturas de transición observadas para C/N , y este aumenta con el campo magnético, siendo máximo en $q = 4$.

Ahora, vamos a mostrar gráficas que comparan las cantidades termodinámicas mostradas pero para $q = 4$ y $B = 1$, con los L usados en la simulación:

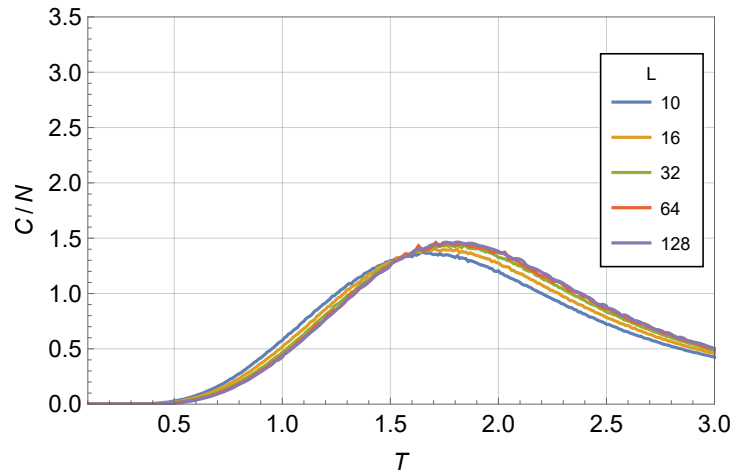


Fig. 2.20: Calor específico C/N en función de T , con $B = 1$ apuntando al eje y , para distintos largos de red L con $q = 4$.

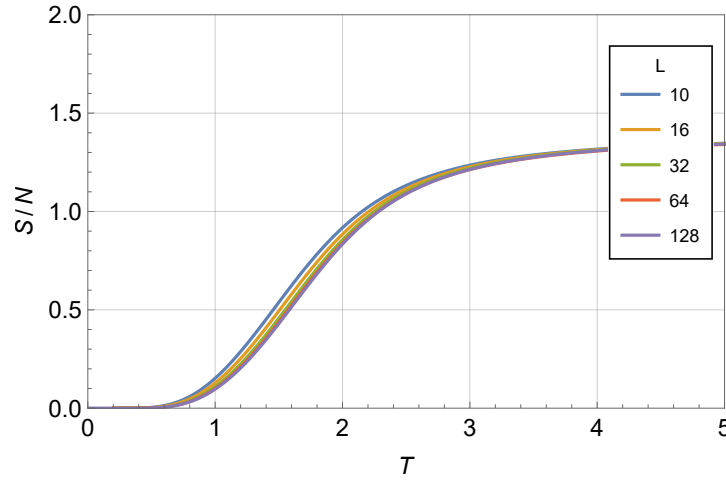


Fig. 2.21: Entropía S/N en función de T , con $B = 1$ apuntando al eje y , para distintos largos de red L con $q = 4$.

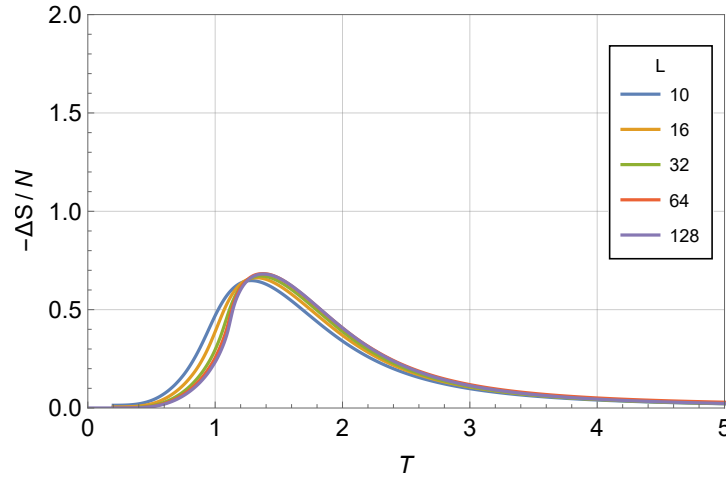


Fig. 2.22: $\Delta S_{iso}/N$ en función de T , con $B = 1$ apuntando al eje y , para distintos largos de red L con $q = 4$.

Viendo estas, podemos ver que el calor específico y la diferencia de entropía aumentan mientras aumentamos el número de espines, mientras que la entropía disminuye, como también el peak es desplazado a temperaturas ligeramente más altas, pero en general, estas diferencias son mínimas y se reducen con el número de espines. Esto es una buena señal ya que nos dice que podemos estudiar sistemas de una gran cantidad de espines analizando que ocurre con sistemas más pequeños, y usando las cantidades normalizadas.

§3 Conclusiones

En este trabajo estudiamos la mecánica estadística y termodinámica de variados sistemas de espines considerando interacciones de exchange entre vecinos y acoplamiento espín-campo magnético, enfocándonos en la obtención de resultados nuevos para un sistema de q -clock 2D y con el objetivo principal de analizar que ocurre con el efecto magnetocalórico en este. Para ello tuvimos que usar distintos métodos analíticos y, en su mayoría, numéricos y computacionales, en especial el método

Monte Carlo con algoritmo Metropoli, usando distintos tamaños de red para poder obtener las cantidades deseadas, normalizarlas por la cantidad de espines, y así analizar estas.

En relación a los resultados, primero logramos observar los resultados esperados para las distintas redes con $q = 2, 4, 6, 8$, en dónde obtuvimos una transición de fase para $q < 5$, con la presencia de un estado ferromagnético ordenado a temperaturas bajas, y un estado paramagnético desordenado a temperaturas altas, mientras que para $q \geq 5$, se observan dos transiciones de fase, una de ellas llamada fase de vórtice o BKT. A temperaturas bajas empezamos en un estado ferromagnético, para luego a temperaturas medias pasar a un estado BKT, caracterizado por formación de vórtices, para terminar a temperaturas altas con un estado paramagnético. El efecto del campo magnético es el de aumentar la temperatura de transición, además de aumentar el ancho de los peaks.

El mayor $-\Delta S_{iso}/N$ se encontró en $q = 4$ con $B = 1$, y el análisis para distintos tamaños de red nos indica que al aumentar este, aumenta el efecto, pero cada vez en magnitudes menores y despreciables luego de 64×64 , por lo que podemos concluir que el análisis de una red de tamaño pequeño nos puede dar resultados aproximados para redes grandes.

§4 Proyecciones a futuro

Una forma de ampliar nuestro modelo es la inclusión de más interacciones que no fueron nombradas, específicamente la presencia de anisotropía en la red, correspondiente a una dirección privilegiada por los espines, y otra posible interacción es la interacción dipolar entre espines, la cuál fue despreciada ya que disminuye en gran cantidad con la distancia.

Estos dos términos afectan en el Hamiltoniano como [25] [26]:

$$H_{anisotropy} = K \sum_i (S_i \cdot \hat{n})^2 \quad (2.1)$$

$$H_{dipolar} = \alpha \sum_{i < j} \frac{S_i \cdot S_j - 3(S_i \cdot \hat{r}_{ij})(S_j \cdot \hat{r}_{ij})}{r_{ij}^3} \quad (2.2)$$

Otra forma de ampliar el trabajo es usar otros tipos de algoritmo para el cálculo Monte Carlo, y comparar resultados. También es posible usar redes con espines faltantes. Finalmente, en este trabajo se podría observar que ocurre con las otras cantidades termodinámicas como energía interna, magnetización y también obtener curvas de histéresis, y realizar un mejor análisis de las transiciones de fase.

Capítulo 3

Anexos

§1 Energía por campo externo

Podemos calcular el trabajo en un inductor considerando un pequeño cambio en el flujo magnético, y aplicando la ley de Faraday [27]:

$$\delta w = -\mathcal{E}I\delta t = I\delta\Phi \quad (3.1)$$

Ahora, con la ley de Ampère:

$$I = \oint \mathbf{H} \cdot d\mathbf{l} \quad (3.2)$$

Podemos reescribir este resultado como, y considerando $\mathbf{B}$ como la inducción magnética, tendremos:

$$\delta w = -\mathcal{E}I\delta t = \oint \delta\Phi \mathbf{H} \cdot d\mathbf{l} = \int \mathbf{H} \cdot \delta\mathbf{B} d^3r \quad (3.3)$$

Cuando la magnetización es uniforme, esta expresión puede ser reescrita como:

$$\delta W = \mathbf{H} \cdot \delta\mathbf{B} \quad (3.4)$$

Con δW el incremento de energía por unidad de volumen.

Ahora, tenemos que el campo magnético $\mathbf{H}$ esta compuesto de 2 contribuciones, un campo magnético externo $\mathbf{H}'$ y un campo demagnetizante, proveniente del material, que es $\mathbf{H}_d$, y la inducción magnética cumple que:

$$\mathbf{B} = \mu_0(\mathbf{H} + \mathbf{M}) = \mu_0(\mathbf{H}' + \mathbf{H}_d + \mathbf{M}) \quad (3.5)$$

Con:

$$\nabla \cdot \mathbf{H}' = 0 \quad (3.6)$$

$$\nabla \times \mathbf{H}' = \mathbf{j}', \quad \text{Con } \mathbf{j}' \text{ una distribución de corriente externa.} \quad (3.7)$$

$$\nabla \cdot \mathbf{H}_d = -\nabla \cdot \mathbf{M} \quad (3.8)$$

$$\nabla \times \mathbf{H}_d = 0 \quad (3.9)$$

Con esto, la expresión de trabajo también contiene la energía del campo magnético $\mathbf{H}$ en un espacio vacío $\int \mu_0 \mathbf{H}' \cdot \delta \mathbf{H}' d^3r$, la cuál no cambia nada la energía del sistema magnético con el que

trabajamos, por lo que la restamos a la expresión original de trabajo. Y con esto, tendremos que:

$$\delta w' = \int \mathbf{H} \cdot \delta \mathbf{B} - \mu_0 \mathbf{H}' \cdot \delta \mathbf{H}' d^3 r \quad (3.10)$$

$$= \mu_0 \left[\int \delta(\mathbf{H}' \cdot \mathbf{H}_d) d^3 r + \int \mathbf{H}_d \cdot \delta \mathbf{H}_d d^3 r + \int_V \mathbf{H} \cdot \delta \mathbf{M} d^3 r \right] \quad (3.11)$$

La primera integral es 0 ya que $\nabla \cdot \mathbf{H}' = 0$, $\nabla \times \mathbf{H}_d = 0$, la segunda integral es la energía propia magnetostática, que la podemos reescribir como:

$$\delta \varepsilon_m = -\mu_0 \int_V \mathbf{H}_d \cdot \delta \mathbf{M} d^3 r \quad (3.12)$$

Y con esto, finalmente tendremos:

$$\delta w' = \int_V \mathbf{H}' \cdot \delta \mathbf{M} d^3 r \quad (3.13)$$

$$\delta W' = \mu_0 \mathbf{H}' \cdot \delta \mathbf{M} \quad (3.14)$$

§2 Ensemble Canónico

Consideramos un sistema cerrado de un reservorio junto a el sistema a analizar, en lo que vamos a tener que la energía total es [13]:

$$E = E_R + E_S \quad (3.15)$$

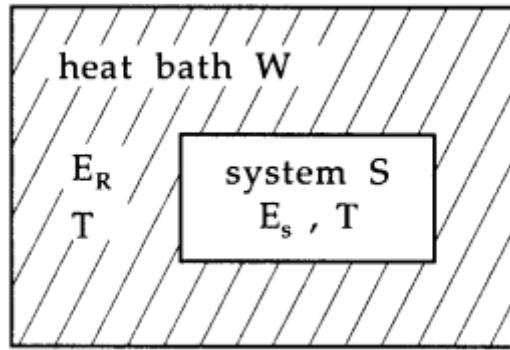


Fig. 3.1: Sistema en un baño térmico/reservorio [13].

Dado que el reservorio es muy grande comparado al sistema en cuestión, tenemos que:

$$\frac{E_S}{E} = \left(1 - \frac{E_R}{E}\right) \ll 1 \quad (3.16)$$

Si S es un sistema cerrado, la probabilidad de encontrar el microestado i con cierta energía E_i , p_i es proporcional a su número de microestados con la energía correspondiente $\Omega_S(E_i)$. También, p_i es proporcional al número de microestados en el sistema total con S en el microestado i y energía E_i , que es igual al número de microestados con energía $E - E_i$:

$$p_i \propto \Omega_R(E_R) = \Omega_R(E - E_i) \quad (3.17)$$

Ahora, por la condición de energías, podemos expandir Ω_R con respecto a E_i , pero primero expandimos la entropía del reservorio $S_R = k_B \ln \Omega_R$:

$$k_B \ln \Omega_R(E - E_i) \approx k_B \ln \Omega_R(E) - E_i \frac{\partial}{\partial E} (k_B \ln \Omega_R(E)) + \dots \quad (3.18)$$

Recordando que tenemos la siguiente relación:

$$\frac{\partial}{\partial E} (k_B \ln \Omega_R(E)) = \frac{\partial S_R}{\partial E} = \frac{1}{T} \quad (3.19)$$

Entonces, 3.18 queda:

$$k_B \ln \Omega_R(E - E_i) = k_B \ln \Omega_R(E) - \frac{E_i}{T} \quad (3.20)$$

$$\ln \Omega_R(E - E_i) \approx \ln \Omega_R(E) - \beta E_i, \quad \beta = \frac{1}{k_B T} \quad (3.21)$$

$$\Omega_R(E - E_i) \approx \Omega_R(E) e^{-\beta E_i} = C e^{-\beta E_i} \quad (3.22)$$

Con C una constante.

Ahora, podemos obtener la probabilidad p_i al también considerar que esta debe ser normalizada con $\sum_i p_i = 1$, con lo que queda:

$$p_i = \frac{e^{-\beta E_i}}{\sum_i e^{-\beta E_i}} \quad (3.23)$$

Denotamos como función de partición:

$$Z = \sum_i e^{-\beta E_i} \quad (3.24)$$

Y la energía interna de nuestro sistema la podemos obtener con:

$$U = \langle E \rangle = \sum_i E_i p_i = \frac{\sum_i E_i e^{-\beta E_i}}{\sum_i e^{-\beta E_i}} = -\frac{\partial}{\partial \beta} \ln \left(\sum_i e^{-\beta E_i} \right) = -\frac{\partial}{\partial \beta} \ln Z \quad (3.25)$$

Comparando la ecuación (1.25) con (3.25), y recordando la definición de β , notamos que:

$$F = -k_B \ln Z \quad (3.26)$$

§3 Códigos computacionales

§3.1 Código C++

El código de C++ fue realizado en su completitud por el prof. Patricio Vargas y Oscar A. Negrete, y fue ocupado para obtener todos los datos de las simulaciones para los distintos casos analizados.

```

25     INFORMACION DE OUTPUT-FILE:
26     1. Temperatura                                7.  Suceptibilidad en X
27     2. Energía Interna                            8.  Suceptibilidad en Y
28     3. Calor Especifico                          9.  Suceptibilidad a partir de MM
29     4. Magnetizacion en X                        10. N° de promedios
30     5. Magnetizacion en Y                        11. N° de configuraciones aceptadas (en
        promedios)
31     6. Magnetizacion MM: sqrt(Mx2 + My2)
32 */
33
34 #include <iostream>                                // Screen output
35 #include <cmath>                                    // Math functions
36 #include <random>                                    // Random functions
37 #include <string>                                    // String manipulations
38 #include <iomanip>                                    // Precicion manipulation
39 #include <fstream>
40 #include <sstream>
41 #include <algorithm>
42 #include <ctime>
43 #include <cstdlib>
44 #include <vector>
45 #include <array>
46
47
48
49
50 // ----- PARAMETROS DE LA SIMULACION ----- //
51 const int    L          = 32;
52 const int    N_spins    = L*L;
53 const int    Q          = 2;           // q=2 --> Ising
54 const double J         = 1.0;         // FM: J>0  AFM: J<0
55
56 const double K         = 0.0;         // Constante de anisotropia

```

```

57 const double alpha      = 0.0;           // Angulo medido en grados!
58
59 std::string ini_State    = "1";           // "Random" or "1", "2", ... , "Q"
60
61 bool use_B_File         = false;
62 bool use_T_File         = false;
63
64 const double B_ini      = 0.0;
65 const double B_end      = 1.2;
66 const double dB         = 0.2;
67
68 const double T_ini      = 0.01;
69 const double T_end      = 5.00;
70 const double dt         = 0.01;
71
72 const int thermLoops    = 10000;          // Numero de loops: thermLoops*(L*L)
73 const int avgLoops      = 500000;        // Numero de promedios: avgLoops*(L*L)
74 long double divNumber   = L*L*avgLoops;
75 // -----
76
77 # define M_PI           3.14159265358979323846 // pi //
78
79
80
81 typedef unsigned long long int Ulong;
82 typedef unsigned int Uint;
83
84 struct RandomGenerator {
85     Ulong v,w;
86     RandomGenerator(Ulong j) : v(4101842887655102017LL), w(1) {
87         v ^= j;
88         w = int64();
89         v = int64();
90     }
91     inline Ulong int64() {
92         v ^= v >> 17; v ^= v << 31; v ^= v >> 8;
93         w = 4294957665U*(w & 0xffffffff) + (w >> 32);

```

```

91     inline Ullong int64() {
92         return v ^ w;
93     }
94     inline double doub() { return 5.42101086242752217E-20 * int64(); }
95     inline Uint int32() { return (Uint)int64(); }
96 };
97
98 RandomGenerator rGenerator(std::time(0));
99
100 template<typename RandomAccessIterator>
101 void knuthShuffle(RandomAccessIterator begin, RandomAccessIterator end) {
102     for(unsigned int n = end - begin - 1; n >= 1; --n) {
103         unsigned int k = rGenerator.int64() % (n + 1);
104         if(k != n) {
105             std::iter_swap(begin + k, begin + n);
106         }
107     }
108 }
109
110 // ----- DECLARACION DE FUNCIONES -----//
111 double dotP(double vec1[], double vec2[]);
112 double energyState(int siteSpin[L+2][L+2], double q_States[Q+1][2], double
    anisotropy[2], double K, double J, double B);
113 void printConfig(int siteSpin[L+2][L+2]);
114 double delta_Energy(int siteSpin[L+2][L+2], double q_States[Q+1][2], double
    S1_vec_old[2], double S1_vec_new[2], int pos_i_S1, int pos_j_S1, double
    anisotropy[2], double K, double J, double B);
115 void getMxMy(int siteSpin[L+2][L+2], double q_States[Q+1][2], double &Mx, double
    &My, double &M);
116 void createIniState(int siteSpin[L+2][L+2], std::string ini_State);
117
118 std::string To_String(double Number, int nDecimals);
119 static int* FisherYatesShuffle(int* data, int count);
120 // -----//

```

```
125 std::vector<double> VEC_Loop01;
126 std::vector<double> VEC_Loop02;
127 std::vector<double> VEC_Loop03;
128
129
130
131
132
133
134
135
136
137
138
139
140 /* *****
141      ***** PROGRAMA PRINCIPAL *****
142      ***** */
143
144 int main() {
145
146     int siteSpin[L+2][L+2];
147     double T, B;
148     double anisotropy[2] = {cos(alpha * M_PI/180), sin(alpha * M_PI/180)};
149     std::string B_print, T_print, nameFile;
150
151     double q_States[Q+1][2];
152     q_States[0][0] = q_States[0][1] = 0;
153     for(int iQ = 1; iQ < Q+1; iQ++ ) {
154         q_States[iQ][0] = sin((iQ-1)*(2*M_PI)/Q);
155         q_States[iQ][1] = cos((iQ-1)*(2*M_PI)/Q);
156     }
157
158     // //////////// START: LECTURA DE ARCHIVO O ARREGLO ////////////
159     if(use_T_File) {
160         std::fstream T_File;
```

```
161     T_File.open("T_File.dat", std::ios_base::in);
162     double T_reader;
163
164     while(T_File.good()) {
165         T_File >> T_reader;
166         VEC_Loop02.push_back(T_reader);
167     }
168
169 } else
170     for(double T = T_ini; T <= T_end; T += dt)
171         VEC_Loop02.push_back(T);
172
173 if(use_B_File) {
174     std::fstream B_File;
175     B_File.open("B_File.dat", std::ios_base::in);
176     double B_reader;
177
178     while(B_File.good()) {
179         B_File >> B_reader;
180         VEC_Loop01.push_back(B_reader);
181     }
182
183 } else
184     for(double B = B_ini; B <= B_end; B += dB)
185         VEC_Loop01.push_back(B);
186 // //////////////////////////////////// END: LECTURA DE ARCHIVO O ARREGLO ////////////////////////////////////
187
188
189 std::vector<double>::iterator pt_Loop01 = VEC_Loop01.begin();
190 std::vector<double>::iterator pt_Loop02 = VEC_Loop02.begin();
191
192 /*
193 Los vectores X e Y contienen, por separado, valores de 1 a L
194 tal que al momento de seleccionar un espin (i,j) se usa X e Y barajados
195 */
```

[illegible]

[illegible]

```

262     double Eini = energyState(siteSpin, q_States, anisotropy, K, J, B);
263
264     /*
265     -----
266         ----- THERMALIZATION LOOP -----
267     -----
268     */
269     for(int trm_counter = 0; trm_counter < thermLoops; trm_counter++) {
270         knuthShuffle(X.begin(),X.end());
271         knuthShuffle(Y.begin(),Y.end());
272
273         for(int l1 = 0; l1 < L; l1++) {
274             for(int l2 = 0; l2 < L; l2++){
275
276                 int p  = X[l1];
277                 int q  = Y[l2];
278
279                 double S1_vec_old[2];
280                 int S1_state_old      = siteSpin[p][q];
281                 S1_vec_old[0]         = q_States[S1_state_old][0];
282                 S1_vec_old[1]         = q_States[S1_state_old][1];
283
284                 double S1_vec_new[2];
285                 int S1_state_new      = 1 + rGenerator.int64()%(Q);
286                 S1_vec_new[0]         = q_States[S1_state_new][0];
287                 S1_vec_new[1]         = q_States[S1_state_new][1];
288
289                 siteSpin[p][q]       = S1_state_new;
290
291                 double deltaE = delta_Energy(siteSpin, q_States,
292                                             S1_vec_old, S1_vec_new, p, q, anisotropy, K, J, B);
293                 double Efini   = Eini + deltaE;

```

```

294         double eta      = rGenerator.doub();
295         double acc_rate = exp(-deltaE/T);
296         double alpha     = eta - acc_rate;
297
298         if(alpha >= 0)
299             siteSpin[p][q] = S1_state_old;
300         else
301             Eini = Efini;
302     }
303 }
304 }
305
306 double Mx_ini = 0, My_ini = 0, MM_ini = sqrt(Mx_ini*Mx_ini +
307         My_ini*My_ini);
308 getMxMy(siteSpin, q_States, Mx_ini, My_ini, MM_ini);
309
310 for(long int d = 0; d < avgLoops; d++) {
311     /*
312     =====
313     ===== SMART SAMPLING =====
314     =====
315     */
316     knuthShuffle(X.begin(),X.end());
317     knuthShuffle(Y.begin(),Y.end());
318
319     for(int l1 = 0; l1 < L; l1++) {
320         for(int l2 = 0; l2 < L; l2++){
321             sum_energies      += Eini;
322             square_sum_energies += Eini*Eini;
323
324             sum_Mx            += Mx_ini;
325             sum_My            += My_ini;

```

```

326         sum_MM           += MM_ini;|
327         sum_square_Mx     += Mx_ini*Mx_ini;
328         sum_square_My     += My_ini*My_ini;
329         sum_square_MM     += MM_ini*MM_ini;
330
331         int p  = X[l1];
332         int q  = Y[l2];
333
334         double S1_vec_old_local[2];
335         int S1_state_old_local    = siteSpin[p][q];
336         S1_vec_old_local[0]      =
337             q_States[S1_state_old_local][0];
338         S1_vec_old_local[1]      =
339             q_States[S1_state_old_local][1];
340
341         double S1_vec_new_local[2];
342         int S1_state_new_local    = 1 +
343             rGenerator.int64()%(Q);
344         S1_vec_new_local[0]      =
345             q_States[S1_state_new_local][0];
346         S1_vec_new_local[1]      =
347             q_States[S1_state_new_local][1];
348
349         siteSpin[p][q] = S1_state_new_local;
350
351         double deltaE = delta_Energy(siteSpin, q_States,
352             S1_vec_old_local, S1_vec_new_local, p, q,
353             anisotropy, K, J, B);
354         double Efini  = Eini + deltaE;
355
356         double deltaMx = S1_vec_new_local[0] -
357             S1_vec_old_local[0];
358         double deltaMy = S1_vec_new_local[1] -

```

```
352         double Mx_fini = Mx_ini + deltaMx;|
353         double My_fini = My_ini + deltaMy;
354         double MM_fini = sqrt(Mx_fini*Mx_fini +
                                My_fini*My_fini);
355
356
357         double eta = rGenerator.doub();
358         double acc_rate = exp(-deltaE/T);
359         double alpha = eta - exp(-deltaE/T);
360
361         if(alpha >= 0)
362             siteSpin[p][q] = S1_state_old_local;
363         else {
364             Eini = Efini;
365             Mx_ini = Mx_fini;
366             My_ini = My_fini;
367             MM_ini = MM_fini;
368             ASP_counter += 1;
369         }
370     }
371 }
372 // ===== END: SMART SAMPLING
373 // =====
374
375 U = sum_energies/divNumber;
376 U2 = square_sum_energies/divNumber;
377 CE = (U2 - U*U)/(T*T);
378
379 Mx = sum_Mx/divNumber;
380 My = sum_My/divNumber;
381 MM = sum_MM/divNumber;
382 Mx2 = sum_square_Mx/divNumber;
383 My2 = sum_square_My/divNumber;
384 MM2 = sum_square_MM/divNumber;
385
```

```

386         X_x = (Mx2 - Mx*Mx)/T;
387         X_y = (My2 - My*My)/T;
388         X_M = (MM2 - MM*MM)/T;
389
390         fprintf(obsFILE, "%f %Lf %Lf %f %f %f %Lf %Lf %Lf %Lf
           %li\n", *pt_Loop02, U, CE, Mx, My, MM, X_x, X_y, X_M, divNumber,
           ASP_counter);
391     }
392     // @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@ END: TEMPERATURE LOOP
       @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
393     fclose(obsFILE);
394 }
395 return 0;
396 }
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414 /* ----- DEFINICION DE FUNCIONES ----- */
415
416 double dotP(double vec1[], double vec2[]) {
417     double sum = 0;
418     for (int i = 0; i < 2; i++)

```

```

419         sum += vec1[i]*vec2[i];|
420     return sum;
421 }
422
423
424 double energyState(int siteSpin[L+2][L+2], double q_States[Q+1][2], double
    anisotropy[2], double K, double J, double B) {
425     double totalEnergy = 0;
426     double S1[2], S2_up[2], S2_right[2];
427     double B_field[2];
428     int stateS1, stateS2_up, stateS2_right;
429
430     B_field[0] = 0;
431     B_field[1] = B;
432
433     for (int i = 1; i < L+1; i++)
434         for (int j = 1; j < L+1; j++) {
435             stateS1      = siteSpin[i][j];
436             stateS2_up   = siteSpin[i-1][j];
437             stateS2_right = siteSpin[i][j+1];
438
439             S1[0]        = q_States[stateS1][0];
440             S1[1]        = q_States[stateS1][1];
441             S2_up[0]     = q_States[stateS2_up][0];
442             S2_up[1]     = q_States[stateS2_up][1];
443             S2_right[0]  = q_States[stateS2_right][0];
444             S2_right[1]  = q_States[stateS2_right][1];
445
446             totalEnergy += -( J * ( dotP(S1,S2_up) + dotP(S1,S2_right) ) +
                dotP(S1,B_field) ) - ( K * dotP(S1, anisotropy)*dotP(S1,
                anisotropy));
447         }
448     return totalEnergy;
449 }
450
451

```

```

452 void printConfig(int siteSpin[L+2][L+2]) {
453     for(int i = 0; i < (L+2); i++) {
454         for(int j = 0; j < (L+2); j++)
455             std::cout << siteSpin[i][j] << " ";
456         std::cout << std::endl;
457     }
458 }
459
460 double delta_Energy(int siteSpin[L+2][L+2], double q_States[Q+1][2], double
    S1_vec_old[2], double S1_vec_new[2], int pos_i_S1, int pos_j_S1, double
    anisotropy[2], double K, double J, double B) {
461     int i = pos_i_S1;
462     int j = pos_j_S1;
463
464     double S2_up[2], S2_down[2], S2_right[2], S2_left[2];
465     int stateS2_up, stateS2_down, stateS2_right, stateS2_left;
466
467     double B_field[2];
468
469     B_field[0] = 0;
470     B_field[1] = B;
471
472     stateS2_up = siteSpin[i-1][j];
473     stateS2_down = siteSpin[i+1][j];
474     stateS2_right = siteSpin[i][j+1];
475     stateS2_left = siteSpin[i][j-1];
476
477     S2_up[0] = q_States[stateS2_up][0];
478     S2_up[1] = q_States[stateS2_up][1];
479     S2_down[0] = q_States[stateS2_down][0];
480     S2_down[1] = q_States[stateS2_down][1];
481     S2_right[0] = q_States[stateS2_right][0];
482     S2_right[1] = q_States[stateS2_right][1];
483     S2_left[0] = q_States[stateS2_left][0];
484     S2_left[1] = q_States[stateS2_left][1];
485

```

```

486     double A[2];|
487     A[0] = S1_vec_new[0] - S1_vec_old[0];
488     A[1] = S1_vec_new[1] - S1_vec_old[1];
489
490     double S_near[2];
491     S_near[0] = S2_up[0] + S2_down[0] + S2_right[0] + S2_left[0];
492     S_near[1] = S2_up[1] + S2_down[1] + S2_right[1] + S2_left[1];
493
494     return -(J * dotP(A,S_near) + dotP(A,B_field)) - K * (dotP(S1_vec_new,
        anisotropy)*dotP(S1_vec_new, anisotropy) - dotP(S1_vec_old,
        anisotropy)*dotP(S1_vec_old, anisotropy));
495 }
496
497
498 void getMxMy(int siteSpin[L+2][L+2], double q_States[Q+1][2], double &Mx, double
    &My, double &MM) {
499     int S1_state = 0;
500     double Mx_local = 0, My_local = 0, Mx2_local = 0, My2_local = 0;
501
502     for (int i = 1; i <= L; i++) {
503         for (int j = 1; j <= L; j++) {
504             S1_state = siteSpin[i][j];
505             Mx_local += q_States[S1_state][0];
506             My_local += q_States[S1_state][1];
507         }
508     }
509
510     Mx += Mx_local;
511     My += My_local;
512     MM += sqrt(Mx_local*Mx_local + My_local*My_local);
513
514     return;
515 }
516
517

```

```
518 void createIniState(int siteSpin[L+2][L+2], std::string ini_State) {
519     for(int i = 0; i < (L+2); i++)
520         for(int j = 0; j < (L+2); j++) {
521             if (i == 0 or i == L+1 or j == 0 or j == L+1)
522                 siteSpin[i][j] = 0;
523             else {
524                 siteSpin[i][j] = (ini_State == "Random") ? (1 +
525                     rGenerator.int64()%(Q)) : std::stoi(ini_State);
526             }
527         }
528     return;
529 }
530
531 std::string To_String(double Number, int nDecimals) {
532     std::ostringstream convert;
533     convert << std::fixed << std::setprecision(nDecimals) << Number;
534     return convert.str();
535 }
536
```

§3.2 Código Mathematica

```
In[*]:= (*LEE DATOS*)
SetDirectory["C:\\Users\\fiur3\\Desktop\\10x10"]

Out[*]:= C:\\Users\\fiur3\\Desktop\\10x10

In[*]:= Datos10x10q2b00 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_0.000_.dat"];
Datos10x10q2b02 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_0.200_.dat"];
Datos10x10q2b04 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_0.400_.dat"];
Datos10x10q2b06 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_0.600_.dat"];
Datos10x10q2b08 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_0.800_.dat"];
Datos10x10q2b10 = Import["dataFM_10x10Q02_K_0.00_A_00.00_B_1.000_.dat"];
Datos10x10q4b00 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_0.000_.dat"];
Datos10x10q4b02 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_0.200_.dat"];
Datos10x10q4b04 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_0.400_.dat"];
Datos10x10q4b06 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_0.600_.dat"];
Datos10x10q4b08 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_0.800_.dat"];
Datos10x10q4b10 = Import["dataFM_10x10Q04_K_0.00_A_00.00_B_1.000_.dat"];
Datos10x10q6b00 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_0.000_.dat"];
Datos10x10q6b02 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_0.200_.dat"];
Datos10x10q6b04 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_0.400_.dat"];
Datos10x10q6b06 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_0.600_.dat"];
Datos10x10q6b08 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_0.800_.dat"];
Datos10x10q6b10 = Import["dataFM_10x10Q06_K_0.00_A_00.00_B_1.000_.dat"];
Datos10x10q8b00 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_0.000_.dat"];
Datos10x10q8b02 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_0.200_.dat"];
Datos10x10q8b04 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_0.400_.dat"];
Datos10x10q8b06 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_0.600_.dat"];
Datos10x10q8b08 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_0.800_.dat"];
Datos10x10q8b10 = Import["dataFM_10x10Q08_K_0.00_A_00.00_B_1.000_.dat"];
```

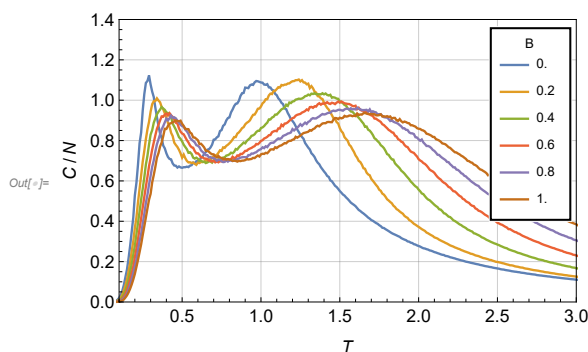
2 | Leedatos 10x10.nb

```

In[ ]:= Datos10x10q2b00N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b00[[All, 1 ;; 3 ;; 2]];
Datos10x10q2b02N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b02[[All, 1 ;; 3 ;; 2]];
Datos10x10q2b04N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b04[[All, 1 ;; 3 ;; 2]];
Datos10x10q2b06N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b06[[All, 1 ;; 3 ;; 2]];
Datos10x10q2b08N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b08[[All, 1 ;; 3 ;; 2]];
Datos10x10q2b10N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q2b10[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b00N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b00[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b02N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b02[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b04N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b04[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b06N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b06[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b08N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b08[[All, 1 ;; 3 ;; 2]];
Datos10x10q4b10N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q4b10[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b00N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b00[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b02N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b02[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b04N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b04[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b06N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b06[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b08N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b08[[All, 1 ;; 3 ;; 2]];
Datos10x10q6b10N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q6b10[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b00N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b00[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b02N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b02[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b04N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b04[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b06N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b06[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b08N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b08[[All, 1 ;; 3 ;; 2]];
Datos10x10q8b10N = {#[[1]], #[[2]] / (10 * 10)} & /@ Datos10x10q8b10[[All, 1 ;; 3 ;; 2]];

In[ ]:= ListLinePlot[{Datos10x10q8b00N, Datos10x10q8b02N, Datos10x10q8b04N, Datos10x10q8b06N,
  Datos10x10q8b08N, Datos10x10q8b10N}, Frame -> True, PlotRange -> {{0.1, 3}, {0, 1.4}},
  PlotLegends -> Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel -> "B",
    LegendFunction -> (Framed[#, FrameMargins -> 0, Background -> White] &),
    LegendMargins -> 5, LabelStyle -> {FontFamily -> "Helvetica", 9}], {0.9, 0.63}],
  Frame -> True, FrameLabel -> {T, HoldForm[C "/" N]},
  LabelStyle -> Directive[Black, FontFamily -> "Helvetica", FontSize -> 12],
  GridLines -> Automatic, FrameTicks -> {{Automatic, None}, {Automatic, None}}]

```

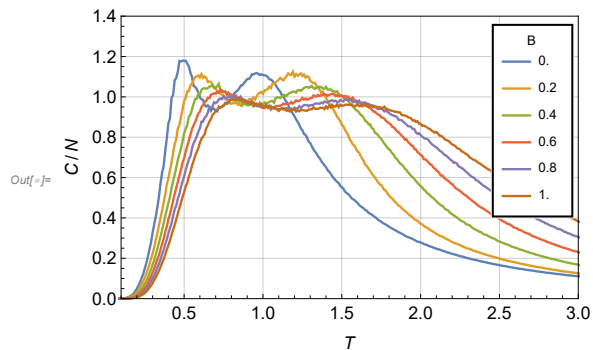


Printed by Wolfram Mathematica Student Edition

```

In[ ]:= ListLinePlot[{Datos10x10q6b00N, Datos10x10q6b02N, Datos10x10q6b04N, Datos10x10q6b06N,
  Datos10x10q6b08N, Datos10x10q6b10N}, Frame → True, PlotRange → {{0.1, 3}, {0, 1.4}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.63}],
  Frame → True, FrameLabel → {T, HoldForm[C "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

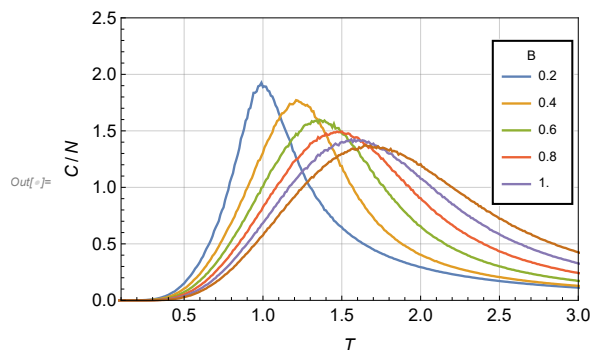
```



```

In[ ]:= ListLinePlot[{Datos10x10q4b00N, Datos10x10q4b02N, Datos10x10q4b04N, Datos10x10q4b06N,
  Datos10x10q4b08N, Datos10x10q4b10N}, Frame → True, PlotRange → {{0.1, 3}, {0, 2.5}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0.2, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.63}],
  Frame → True, FrameLabel → {T, HoldForm[C "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

```

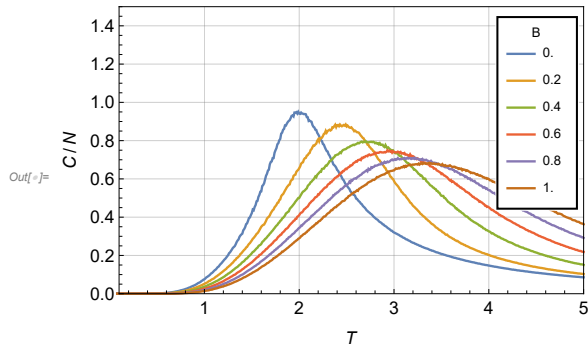


4 | Leedatos 10x10.nb

```

In[ ]:= ListLinePlot[{Datos10x10q2b00N, Datos10x10q2b02N, Datos10x10q2b04N, Datos10x10q2b06N,
  Datos10x10q2b08N, Datos10x10q2b10N}, Frame → True, PlotRange → {{0.1, 5}, {0, 1.5}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.63}],
  Frame → True, FrameLabel → {T, HoldForm[C "/ N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

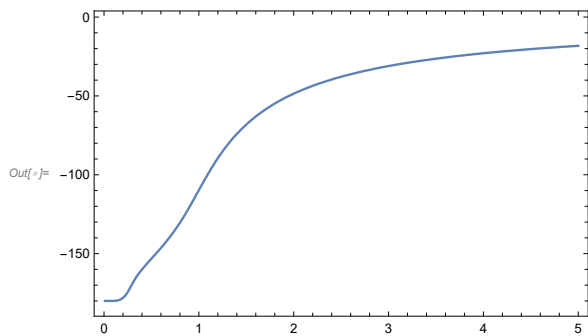
```



```

In[ ]:= ListLinePlot[Datos10x10q8b00[All, 1 ;; 2 ;; 1], Frame → True]

```



```

In[ ]:= CC10x10q2b00N = Interpolation[Datos10x10q2b00N, InterpolationOrder -> 1];
CC10x10q2b02N = Interpolation[Datos10x10q2b02N, InterpolationOrder -> 1];
CC10x10q2b04N = Interpolation[Datos10x10q2b04N, InterpolationOrder -> 1];
CC10x10q2b06N = Interpolation[Datos10x10q2b06N, InterpolationOrder -> 1];
CC10x10q2b08N = Interpolation[Datos10x10q2b08N, InterpolationOrder -> 1];
CC10x10q2b10N = Interpolation[Datos10x10q2b10N, InterpolationOrder -> 1];
CC10x10q4b00N = Interpolation[Datos10x10q4b00N, InterpolationOrder -> 1];
CC10x10q4b02N = Interpolation[Datos10x10q4b02N, InterpolationOrder -> 1];
CC10x10q4b04N = Interpolation[Datos10x10q4b04N, InterpolationOrder -> 1];
CC10x10q4b06N = Interpolation[Datos10x10q4b06N, InterpolationOrder -> 1];
CC10x10q4b08N = Interpolation[Datos10x10q4b08N, InterpolationOrder -> 1];
CC10x10q4b10N = Interpolation[Datos10x10q4b10N, InterpolationOrder -> 1];
CC10x10q6b00N = Interpolation[Datos10x10q6b00N, InterpolationOrder -> 1];
CC10x10q6b02N = Interpolation[Datos10x10q6b02N, InterpolationOrder -> 1];
CC10x10q6b04N = Interpolation[Datos10x10q6b04N, InterpolationOrder -> 1];
CC10x10q6b06N = Interpolation[Datos10x10q6b06N, InterpolationOrder -> 1];
CC10x10q6b08N = Interpolation[Datos10x10q6b08N, InterpolationOrder -> 1];
CC10x10q6b10N = Interpolation[Datos10x10q6b10N, InterpolationOrder -> 1];
CC10x10q8b00N = Interpolation[Datos10x10q8b00N, InterpolationOrder -> 1];
CC10x10q8b02N = Interpolation[Datos10x10q8b02N, InterpolationOrder -> 1];
CC10x10q8b04N = Interpolation[Datos10x10q8b04N, InterpolationOrder -> 1];
CC10x10q8b06N = Interpolation[Datos10x10q8b06N, InterpolationOrder -> 1];
CC10x10q8b08N = Interpolation[Datos10x10q8b08N, InterpolationOrder -> 1];
CC10x10q8b10N = Interpolation[Datos10x10q8b10N, InterpolationOrder -> 1];

In[ ]:= SSN10x10q2b00[T_] :=
  Quiet[NIntegrate[CC10x10q2b00N[x] / x, {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q2b02[T_] := Quiet[NIntegrate[CC10x10q2b02N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q2b04[T_] := Quiet[NIntegrate[CC10x10q2b04N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q2b06[T_] := Quiet[NIntegrate[CC10x10q2b06N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q2b08[T_] := Quiet[NIntegrate[CC10x10q2b08N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q2b10[T_] := Quiet[NIntegrate[CC10x10q2b10N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q4b00[T_] := Quiet[NIntegrate[CC10x10q4b00N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q4b02[T_] := Quiet[NIntegrate[CC10x10q4b02N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q4b04[T_] := Quiet[NIntegrate[CC10x10q4b04N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q4b06[T_] := Quiet[NIntegrate[CC10x10q4b06N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];
SSN10x10q4b08[T_] := Quiet[NIntegrate[CC10x10q4b08N[x] / x,
  {x, 0.05, T}, Method -> "LocalAdaptive"]];

```

```

SSN10x10q4b10[T_] := Quiet[NIntegrate[CC10x10q4b10N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b00[T_] := Quiet[NIntegrate[CC10x10q6b00N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b02[T_] := Quiet[NIntegrate[CC10x10q6b02N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b04[T_] := Quiet[NIntegrate[CC10x10q6b04N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b06[T_] := Quiet[NIntegrate[CC10x10q6b06N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b08[T_] := Quiet[NIntegrate[CC10x10q6b08N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q6b10[T_] := Quiet[NIntegrate[CC10x10q6b10N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b00[T_] := Quiet[NIntegrate[CC10x10q8b00N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b02[T_] := Quiet[NIntegrate[CC10x10q8b02N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b04[T_] := Quiet[NIntegrate[CC10x10q8b04N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b06[T_] := Quiet[NIntegrate[CC10x10q8b06N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b08[T_] := Quiet[NIntegrate[CC10x10q8b08N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];
SSN10x10q8b10[T_] := Quiet[NIntegrate[CC10x10q8b10N[x] / x,
  {x, 0.05, T}, Method → "LocalAdaptive"]];

```

```

In[*]:= SN10x10q2b00 = ParallelTable[{T, SSN10x10q2b00[T] + Log[2] / (10 * 10)}, {T, 0.05, 5, 0.01}];
SN10x10q2b02 = ParallelTable[{T, SSN10x10q2b02[T]}, {T, 0.05, 5, 0.01}];
SN10x10q2b04 = ParallelTable[{T, SSN10x10q2b04[T]}, {T, 0.05, 5, 0.01}];
SN10x10q2b06 = ParallelTable[{T, SSN10x10q2b06[T]}, {T, 0.05, 5, 0.01}];
SN10x10q2b08 = ParallelTable[{T, SSN10x10q2b08[T]}, {T, 0.05, 5, 0.01}];
SN10x10q2b10 = ParallelTable[{T, SSN10x10q2b10[T]}, {T, 0.05, 5, 0.01}];
SN10x10q4b00 = ParallelTable[{T, SSN10x10q4b00[T] + Log[4] / (10 * 10)}, {T, 0.05, 5, 0.01}];
SN10x10q4b02 = ParallelTable[{T, SSN10x10q4b02[T]}, {T, 0.05, 5, 0.01}];
SN10x10q4b04 = ParallelTable[{T, SSN10x10q4b04[T]}, {T, 0.05, 5, 0.01}];
SN10x10q4b06 = ParallelTable[{T, SSN10x10q4b06[T]}, {T, 0.05, 5, 0.01}];
SN10x10q4b08 = ParallelTable[{T, SSN10x10q4b08[T]}, {T, 0.05, 5, 0.01}];
SN10x10q4b10 = ParallelTable[{T, SSN10x10q4b10[T]}, {T, 0.05, 5, 0.01}];
SN10x10q6b00 = ParallelTable[{T, SSN10x10q6b00[T] + Log[6] / (10 * 10)}, {T, 0.05, 5, 0.01}];
SN10x10q6b02 = ParallelTable[{T, SSN10x10q6b02[T]}, {T, 0.05, 5, 0.01}];
SN10x10q6b04 = ParallelTable[{T, SSN10x10q6b04[T]}, {T, 0.05, 5, 0.01}];
SN10x10q6b06 = ParallelTable[{T, SSN10x10q6b06[T]}, {T, 0.05, 5, 0.01}];
SN10x10q6b08 = ParallelTable[{T, SSN10x10q6b08[T]}, {T, 0.05, 5, 0.01}];
SN10x10q6b10 = ParallelTable[{T, SSN10x10q6b10[T]}, {T, 0.05, 5, 0.01}];
SN10x10q8b00 = ParallelTable[{T, SSN10x10q8b00[T] + Log[8] / (10 * 10)}, {T, 0.05, 5, 0.01}];
SN10x10q8b02 = ParallelTable[{T, SSN10x10q8b02[T]}, {T, 0.05, 5, 0.01}];
SN10x10q8b04 = ParallelTable[{T, SSN10x10q8b04[T]}, {T, 0.05, 5, 0.01}];
SN10x10q8b06 = ParallelTable[{T, SSN10x10q8b06[T]}, {T, 0.05, 5, 0.01}];
SN10x10q8b08 = ParallelTable[{T, SSN10x10q8b08[T]}, {T, 0.05, 5, 0.01}];
SN10x10q8b10 = ParallelTable[{T, SSN10x10q8b10[T]}, {T, 0.05, 5, 0.01}];

```

```

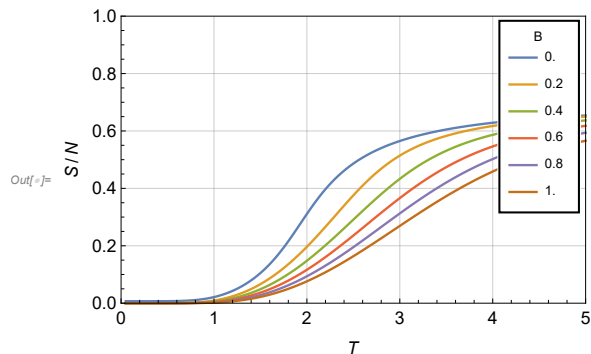
In[*]:= ΔSN10x10q2b02 =
  Transpose[{SN10x10q2b00[[A11, 1]], Subtract[SN10x10q2b00[[A11, 2]], SN10x10q2b02[[A11, 2]]]}];
ΔSN10x10q2b04 = Transpose[
  {SN10x10q2b00[[A11, 1]], Subtract[SN10x10q2b00[[A11, 2]], SN10x10q2b04[[A11, 2]]]}];
ΔSN10x10q2b06 = Transpose[{SN10x10q2b00[[A11, 1]],
  Subtract[SN10x10q2b00[[A11, 2]], SN10x10q2b06[[A11, 2]]]}];
ΔSN10x10q2b08 = Transpose[{SN10x10q2b00[[A11, 1]],
  Subtract[SN10x10q2b00[[A11, 2]], SN10x10q2b08[[A11, 2]]]}];
ΔSN10x10q2b10 = Transpose[{SN10x10q2b00[[A11, 1]],
  Subtract[SN10x10q2b00[[A11, 2]], SN10x10q2b10[[A11, 2]]]}];
ΔSN10x10q8b02 = Transpose[{SN10x10q8b00[[A11, 1]],
  Subtract[SN10x10q8b00[[A11, 2]], SN10x10q8b02[[A11, 2]]]}];
ΔSN10x10q8b04 = Transpose[{SN10x10q8b00[[A11, 1]],
  Subtract[SN10x10q8b00[[A11, 2]], SN10x10q8b04[[A11, 2]]]}];
ΔSN10x10q8b06 = Transpose[{SN10x10q8b00[[A11, 1]],
  Subtract[SN10x10q8b00[[A11, 2]], SN10x10q8b06[[A11, 2]]]}];
ΔSN10x10q8b08 = Transpose[{SN10x10q8b00[[A11, 1]],
  Subtract[SN10x10q8b00[[A11, 2]], SN10x10q8b08[[A11, 2]]]}];
ΔSN10x10q8b10 = Transpose[{SN10x10q8b00[[A11, 1]],
  Subtract[SN10x10q8b00[[A11, 2]], SN10x10q8b10[[A11, 2]]]}];
ΔSN10x10q6b02 = Transpose[{SN10x10q6b00[[A11, 1]],
  Subtract[SN10x10q6b00[[A11, 2]], SN10x10q6b02[[A11, 2]]]}];
ΔSN10x10q6b04 = Transpose[{SN10x10q6b00[[A11, 1]],
  Subtract[SN10x10q6b00[[A11, 2]], SN10x10q6b04[[A11, 2]]]}];
ΔSN10x10q6b06 = Transpose[{SN10x10q6b00[[A11, 1]],
  Subtract[SN10x10q6b00[[A11, 2]], SN10x10q6b06[[A11, 2]]]}];
ΔSN10x10q6b08 = Transpose[{SN10x10q6b00[[A11, 1]],
  Subtract[SN10x10q6b00[[A11, 2]], SN10x10q6b08[[A11, 2]]]}];
ΔSN10x10q6b10 = Transpose[{SN10x10q6b00[[A11, 1]],
  Subtract[SN10x10q6b00[[A11, 2]], SN10x10q6b10[[A11, 2]]]}];
ΔSN10x10q4b02 = Transpose[{SN10x10q4b00[[A11, 1]],
  Subtract[SN10x10q4b00[[A11, 2]], SN10x10q4b02[[A11, 2]]]}];
ΔSN10x10q4b04 = Transpose[{SN10x10q4b00[[A11, 1]],
  Subtract[SN10x10q4b00[[A11, 2]], SN10x10q4b04[[A11, 2]]]}];
ΔSN10x10q4b06 = Transpose[{SN10x10q4b00[[A11, 1]],
  Subtract[SN10x10q4b00[[A11, 2]], SN10x10q4b06[[A11, 2]]]}];
ΔSN10x10q4b08 = Transpose[{SN10x10q4b00[[A11, 1]],
  Subtract[SN10x10q4b00[[A11, 2]], SN10x10q4b08[[A11, 2]]]}];
ΔSN10x10q4b10 = Transpose[{SN10x10q4b00[[A11, 1]],
  Subtract[SN10x10q4b00[[A11, 2]], SN10x10q4b10[[A11, 2]]]}];

```

```

In[ ]:= ListLinePlot[{SN10x10q2b00, SN10x10q2b02, SN10x10q2b04, SN10x10q2b06,
  SN10x10q2b08, SN10x10q2b10}, Frame → True, PlotRange → {{0, 5}, {0, 1}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.65}],
  Frame → True, FrameLabel → {T, HoldForm[S "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

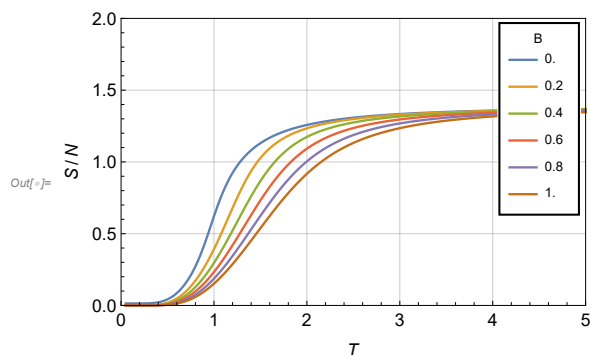
```



```

In[ ]:= ListLinePlot[{SN10x10q4b00, SN10x10q4b02, SN10x10q4b04, SN10x10q4b06,
  SN10x10q4b08, SN10x10q4b10}, Frame → True, PlotRange → {{0, 5}, {0, 2}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.65}],
  Frame → True, FrameLabel → {T, HoldForm[S "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

```

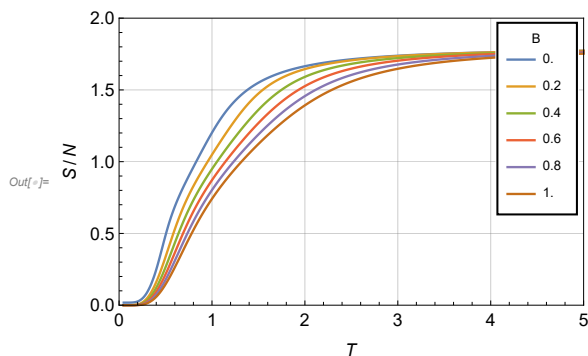


10 | Leedatos 10x10.nb

```

In[ ]:= ListLinePlot[{SN10x10q6b00, SN10x10q6b02, SN10x10q6b04, SN10x10q6b06,
  SN10x10q6b08, SN10x10q6b10}, Frame → True, PlotRange → {{0, 5}, {0, 2}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.65}],
  Frame → True, FrameLabel → {T, HoldForm[S "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

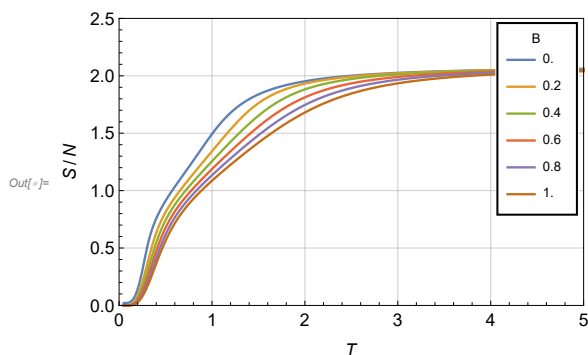
```



```

In[ ]:= ListLinePlot[{SN10x10q8b00, SN10x10q8b02, SN10x10q8b04, SN10x10q8b06,
  SN10x10q8b08, SN10x10q8b10}, Frame → True, PlotRange → {{0, 5}, {0, 2.5}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.65}],
  Frame → True, FrameLabel → {T, HoldForm[S "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

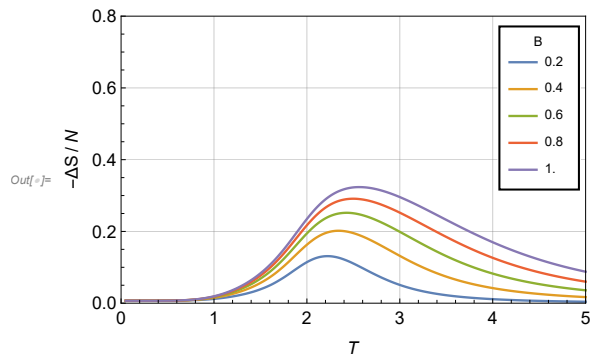
```



```

In[ ]:= ListLinePlot[{ΔSN10x10q2b02, ΔSN10x10q2b04, ΔSN10x10q2b06, ΔSN10x10q2b08, ΔSN10x10q2b10},
  Frame → True, PlotRange → {{0, 5}, {0, 0.8}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0.2, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.68}],
  Frame → True, FrameLabel → {T, HoldForm[-ΔS "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

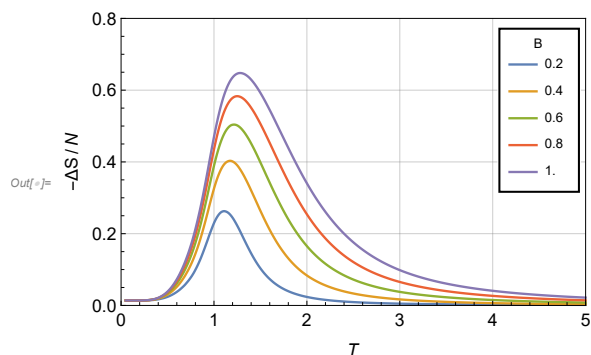
```



```

In[ ]:= ListLinePlot[{ΔSN10x10q4b02, ΔSN10x10q4b04, ΔSN10x10q4b06, ΔSN10x10q4b08, ΔSN10x10q4b10},
  Frame → True, PlotRange → {{0, 5}, {0, 0.8}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0.2, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.68}],
  Frame → True, FrameLabel → {T, HoldForm[-ΔS "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

```

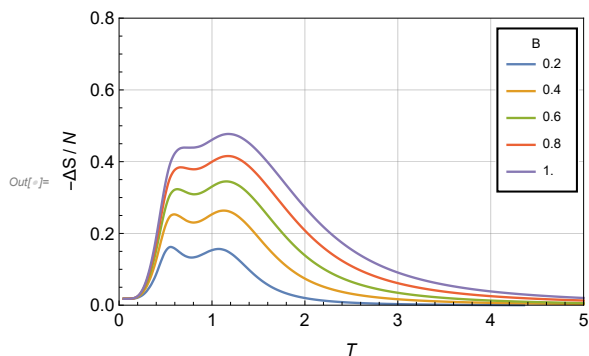


12 | Leedatos 10x10.nb

```

In[ ]:= ListLinePlot[{ΔSN10x10q6b02, ΔSN10x10q6b04, ΔSN10x10q6b06, ΔSN10x10q6b08, ΔSN10x10q6b10},
  Frame → True, PlotRange → {{0, 5}, {0, 0.8}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0.2, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.68}],
  Frame → True, FrameLabel → {T, HoldForm[-ΔS "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

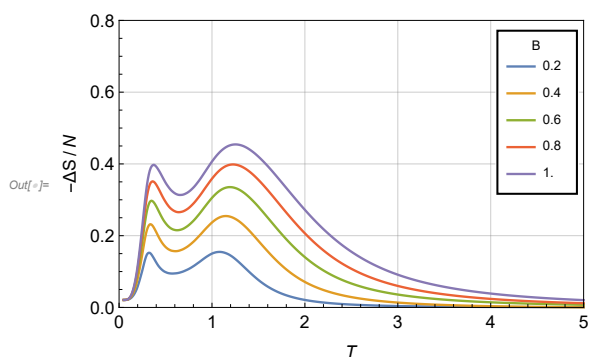
```



```

In[ ]:= ListLinePlot[{ΔSN10x10q8b02, ΔSN10x10q8b04, ΔSN10x10q8b06, ΔSN10x10q8b08, ΔSN10x10q8b10},
  Frame → True, PlotRange → {{0, 5}, {0, 0.8}},
  PlotLegends → Placed[LineLegend[Evaluate[Table[B, {B, 0.2, 1, 0.2}]], LegendLabel → "B",
    LegendFunction → (Framed[#, FrameMargins → 0, Background → White] &),
    LegendMargins → 5, LabelStyle → {FontFamily → "Helvetica", 9}], {0.9, 0.68}],
  Frame → True, FrameLabel → {T, HoldForm[-ΔS "/" N]},
  LabelStyle → Directive[Black, FontFamily → "Helvetica", FontSize → 12],
  GridLines → Automatic, FrameTicks → {{Automatic, None}, {Automatic, None}}]

```



Este corresponde a uno de los códigos usados para obtener los gráficos en Mathematica, comenzando por la importación de los datos, obtención de las gráficas de calor específico, y de estas al interpolar y posteriormente integrar estas, obtenemos las gráficas de entropía.

Capítulo 4

Bibliografía

- [1] J Romero Gómez, R Ferreiro Garcia, A De Miguel Catoira, and M Romero Gómez. Magnetocaloric effect: A review of the thermodynamic cycles in magnetic refrigeration. *Renewable and Sustainable Energy Reviews*, 17:74–82, 2013.
- [2] Emil Warburg. Magnetische untersuchungen. *Annalen der Physik*, 249(5):141–164, 1881.
- [3] Peter Debye. Einige bemerkungen zur magnetisierung bei tiefer temperatur. *Annalen der Physik*, 386(25):1154–1160, 1926.
- [4] WF Giaque. A thermodynamic treatment of certain magnetic effects. a proposed method of producing temperatures considerably below 1 absolute. *Journal of the American Chemical Society*, 49(8):1864–1870, 1927.
- [5] Karl A Gschneidner, VK Pecharsky, and AO Tsokol. Recent developments in magnetocaloric materials. *Reports on progress in physics*, 68(6):1479, 2005.
- [6] Ali Alahmer, Malik Al-Amayreh, Ahmad O Mostafa, Mohammad Al-Dabbas, and Hegazy Rezk. Magnetic refrigeration design technologies: State of the art and general perspectives. *Energies*, 14(15):4662, 2021.
- [7] VK Pecharsky, KA Gschneidner Jr, AO Pecharsky, and AM Tishin. Thermodynamics of the magnetocaloric effect. *Physical review B*, 64(14):144406, 2001.
- [8] NA De Oliveira and Pedro Jorge von RANKE. Theoretical aspects of the magnetocaloric effect. *Physics Reports*, 489(4-5):89–159, 2010.
- [9] Oscar Negrete. *Propiedades Magnetocalóricas De Materiales Altamente Frustrados Con Estructuras tipo Fe₂-P*. Universidad Técnica Federico Santa María, 2018.
- [10] Aleksandr M Tishin and Youry I Spichkin. *The magnetocaloric effect and its applications*. CRC Press, 2016.
- [11] Raj Kumar Pathria. *Statistical mechanics*. Elsevier, 2016.
- [12] Vitalij K Pecharsky and Karl A Gschneidner Jr. Magnetocaloric effect and magnetic refrigeration. *Journal of magnetism and magnetic materials*, 200(1-3):44–56, 1999.
- [13] Walter Greiner, Ludwig Neise, and Horst Stöcker. *Thermodynamics and statistical mechanics*. Springer Science & Business Media, 2012.

- [14] Franz Utermohlen. Mean field theory solution of the ising model, 2018.
- [15] Jozef Strecka and Michal Jascur. A brief account of the ising and ising-like models: Mean-field, effective-field and exact results. *arXiv preprint arXiv:1511.03031*, 2015.
- [16] Michael Tinkham. *Introduction to superconductivity*. Courier Corporation, 2004.
- [17] Cosimo Lupo and Federico Ricci-Tersenghi. Approximating the xy model on a random graph with a q-state clock model. *Physical Review B*, 95(5):054433, 2017.
- [18] Oscar A Negrete, Patricio Vargas, Francisco J Peña, Gonzalo Saravia, and Eugenio E Vogel. Entropy and mutability for the q-state clock model in small systems. *Entropy*, 20(12):933, 2018.
- [19] John Michael Kosterlitz and David James Thouless. Ordering, metastability and phase transitions in two-dimensional systems. *Journal of Physics C: Solid State Physics*, 6(7):1181, 1973.
- [20] G Ortiz, E Cobanera, and Z Nussinov. Dualities and the phase diagram of the p-clock model. *Nuclear Physics B*, 854(3):780–814, 2012.
- [21] Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.
- [22] David Landau and Kurt Binder. *A guide to Monte Carlo simulations in statistical physics*. Cambridge university press, 2021.
- [23] Mark EJ Newman and Gerard T Barkema. *Monte Carlo methods in statistical physics*. Clarendon Press, 1999.
- [24] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.
- [25] M Knobel, LC Sampaio, EHCP Sinnecker, P Vargas, and D Altbir. Dipolar magnetic interactions among magnetic microwires. *Journal of magnetism and magnetic materials*, 249(1-2):60–72, 2002.
- [26] Randy S Fishman. Single-ion anisotropy is necessary and appropriate to study the magnetic behavior of tb 3+ moments with j eff= 1 2 on the honeycomb lattice in tb 2 ir 3 ga 9. *Physical Review B*, 103(21):214440, 2021.
- [27] John MD Coey. *Magnetism and magnetic materials*. Cambridge university press, 2010.