Simulação do Modelo de Ising 2D

Grupo 8

Iremos ocultar nesse PDF o código que já foi demonstrado na tarefa passada. Fique claro, entretanto, que as funções referenciadas aqui são as mesmas definidas anteriormente.

Primeiramente iremos definir algumas funções auxiliares e os intervalos de tamanho e temperatura:

In [49]:
```
L = [18,24,36]
temp = np.linspace(5, 1, 17)

def somar_por_caixa(lista,n_caixas=10):
    m = len(lista)/n_caixas
    lista_somas = list()
    for i in range(n_caixas):
        soma = 0
        a = int(i*m)
        b = int((i+1)*m - 1)
        lista_somas.append(np.sum(lista[a:b])/m)
    return lista_somas

def somar_por_caixa_quadrado(lista,n_caixas=10):
    return somar_por_caixa([x*x for x in lista], n_caixas)

def obter_erros(lista_valores, media, n = 10):
    soma = 0
    for err in lista_valores:
        soma += (media - err)**2
        soma = soma/(n*(n-1))
        soma = np.sqrt(soma)
    return soma
```

Execução

Iremos executar as simulações e calcular todas as grandezas que iremos analisar. Os cálculos são bem simples e são calculados exatamente como orientado no PDF da tarefa. As grandezas são guardadas em um dicionário para posteriormente serem usadas para plotarmos os gráficos necessários.

In [84]:
```
n_iter = 1100000
from tqdm import tqdm

values = dict()
errors = dict()
for tam in L:
    print("Tamanho: ", tam)

    # Listas
    lista_energia      = list()
    lista_energia_erro = list()
    lista_mag          = list()
    lista_mag_erro     = list()
    lista_calor        = list()
    lista_calor_erro   = list()
    lista_sus          = list()
    lista_sus_erro     = list()

    s, viz = gen_spin(tam)
```

```
    for t in tqdm(temp):
        E, mag, beta = calculate_variables(t, s, viz)
        e_list, m_list, s = calculate_energy_and_mag(E, mag, s, beta, viz, n_

        e_list = e_list[int(n_iter/11):]
        m_list = m_list[int(n_iter/11):]

        e_list_p_spin = [x / (tam*tam) for x in e_list]
        m_list_p_spin = [x / (tam*tam) for x in m_list]

        # Calcular Energia----------------------------------------------------
        lista_somas = somar_por_caixa(e_list_p_spin)
        s_energia_p_spin_media = np.mean(lista_somas)

        lista_energia.append(s_energia_p_spin_media)
        energia_erro = obter_erros(lista_somas,s_energia_p_spin_media)
        lista_energia_erro.append(energia_erro*s_energia_p_spin_media)

        # Calcular Magnetismo ------------------------------------------------
        lista_somas = somar_por_caixa(m_list_p_spin)
        s_mag_p_spin_media = np.mean(lista_somas)
        lista_mag.append(s_mag_p_spin_media)
        mag_erro = obter_erros(lista_somas,s_mag_p_spin_media)
        lista_mag_erro.append(mag_erro)

        # Calcular Calor Específico ------------------------------------------
        lista_somas_1 = somar_por_caixa(e_list)
        lista_somas_2 = somar_por_caixa_quadrado(e_list)
        lista_nova = list()
        for i in range(len(lista_somas_1)):
            lista_nova.append((lista_somas_2[i] - (lista_somas_1[i]**2)) * ((

        calor_medio = np.mean(lista_nova)
        lista_calor.append(calor_medio)
        c_erro = obter_erros(lista_nova,calor_medio)
        lista_calor_erro.append(c_erro)

        # Calcular Susceptibilidade Magnética --------------------------------
        lista_somas_1 = somar_por_caixa(m_list)
        lista_somas_2 = somar_por_caixa_quadrado(m_list)
        lista_nova = list()
        for i in range(len(lista_somas_1)):
            lista_nova.append((lista_somas_2[i] - (lista_somas_1[i]**2)) * ((

        sus_medio = np.mean(lista_nova)
        lista_sus.append(sus_medio)
        sus_erro = obter_erros(lista_nova,calor_medio)
        lista_sus_erro.append(sus_erro)

    values[tam] = dict()
    errors[tam] = dict()

    values[tam]["E"] = lista_energia
    values[tam]["M"] = [abs(x) for x in lista_mag]
    values[tam]["C"] = lista_calor
    values[tam]["S"] = lista_sus

    errors[tam]["E"] = lista_energia_erro
    errors[tam]["M"] = lista_mag_erro
    errors[tam]["C"] = lista_calor_erro
    errors[tam]["S"] = lista_sus_erro
```

```
  0%|          | 0/17 [00:00<?, ?it/s]
Tamanho:  18
100%|██████████| 17/17 [01:59<00:00,  7.02s/it]
```

```
  0%|          | 0/17 [00:00<?, ?it/s]
Tamanho:  24
100%|██████████| 17/17 [03:07<00:00, 11.00s/it]
  0%|          | 0/17 [00:00<?, ?it/s]
Tamanho:  36
100%|██████████| 17/17 [06:23<00:00, 22.58s/it]
```

Análise

Vamos agora analisar os gráficos para as seguintes grandezas:

- Energia por Spin
- Módulo da Magnetização por Spin
- Calor Específico
- Susceptibilidade Magnética

Iremos plotar dois gráficos para cada. O primeiro com as médias e com uma barra mostrando os erros e o segundo será o gráfico apenas com os erros. Perceba que em alguns casos, devido à diferença da ordem de grandeza entre os valores e os erros, o tamanho da barra de erro será pequeno ao ponto de ficar menor do que os símbolos. Isso é esperado e já foi discutido com o professor.

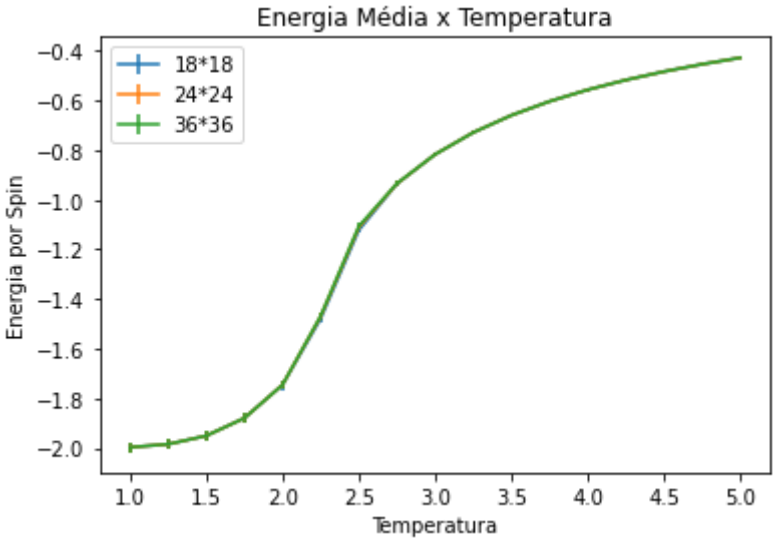
Energia por Spin

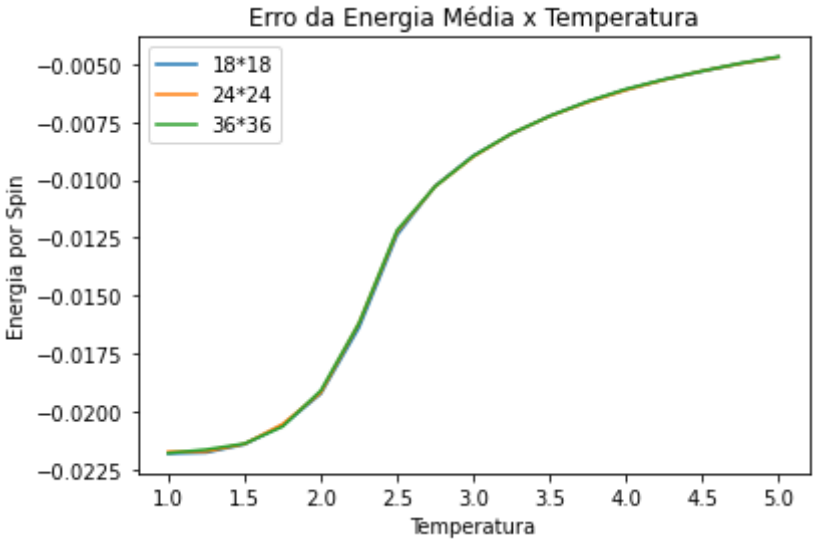
In [109…

```python
# Energia
plt.title("Energia Média x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Energia por Spin")
for tam in L:
    plt.errorbar(temp, values[tam]["E"], yerr = errors[tam]["E"], label = (st
plt.legend()
plt.show()
plt.close()

# Obs: As energias foram muito semelhantes nas redes então as linhas acabaram

plt.title("Erro da Energia Média x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Energia por Spin")
for tam in L:
    plt.plot(temp, errors[tam]["E"], label = (str(tam)+"*"+str(tam)))
plt.legend()
plt.show()
plt.close()
```





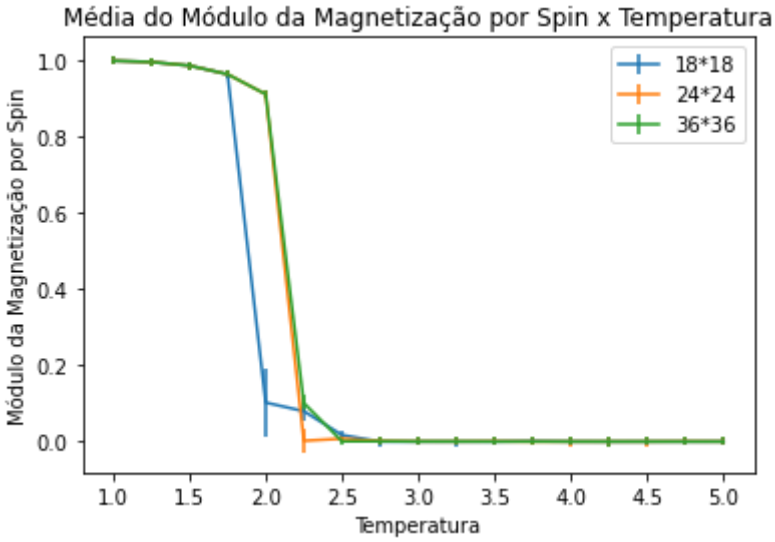
Módulo da Magnetização por Spin

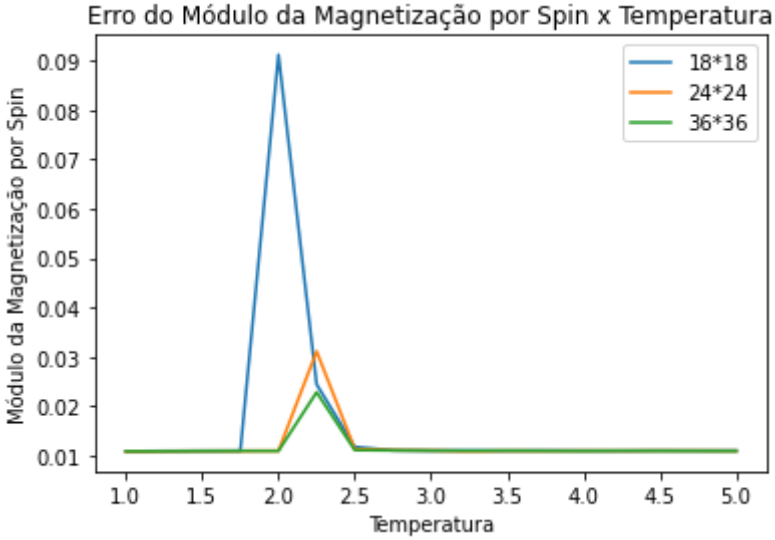
In [111…

```
# Energia
plt.title("Média do Módulo da Magnetização por Spin x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Módulo da Magnetização por Spin")
for tam in L:
    plt.errorbar(temp, values[tam]["M"], yerr = errors[tam]["M"], label = (st
plt.legend()
plt.show()
plt.close()

# Obs: As energias foram muito semelhantes nas redes então as linhas acabaram

plt.title("Erro do Módulo da Magnetização por Spin x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Módulo da Magnetização por Spin")
for tam in L:
    plt.plot(temp, errors[tam]["M"], label = (str(tam)+"*"+str(tam)))
plt.legend()
plt.show()
plt.close()
```





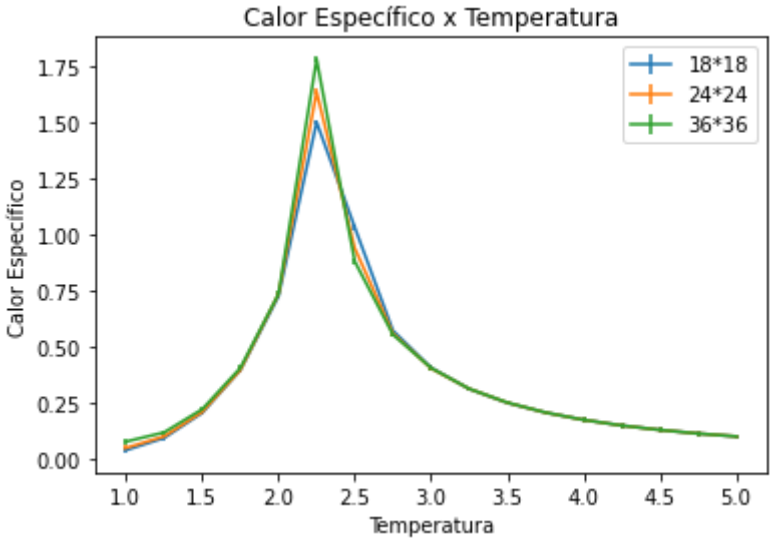
Calor Específico

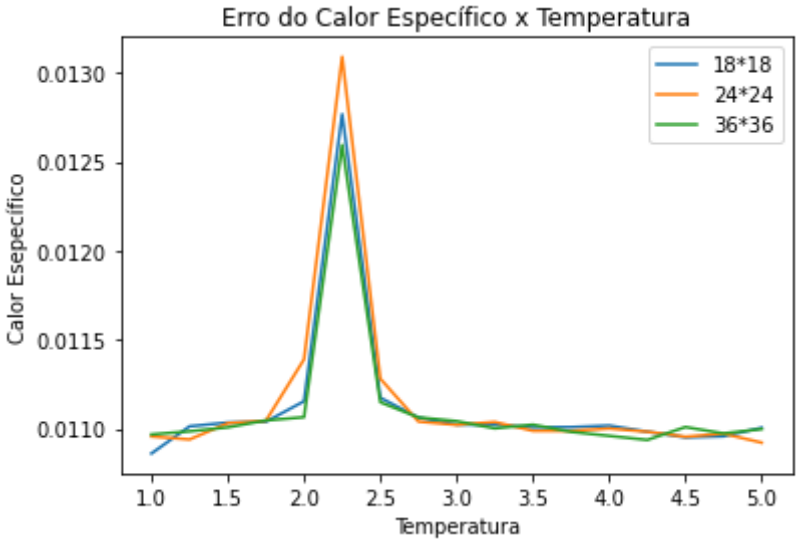
In [113…

```
# Energia
plt.title("Calor Específico x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Calor Específico")
for tam in L:
    plt.errorbar(temp, values[tam]["C"], yerr = errors[tam]["C"], label = (st
plt.legend()
plt.show()
plt.close()

# Obs: As energias foram muito semelhantes nas redes então as linhas acabaram

plt.title("Erro do Calor Específico x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Calor Esepecífico")
for tam in L:
    plt.plot(temp, errors[tam]["C"], label = (str(tam)+"*"+str(tam)))
plt.legend()
plt.show()
plt.close()
```





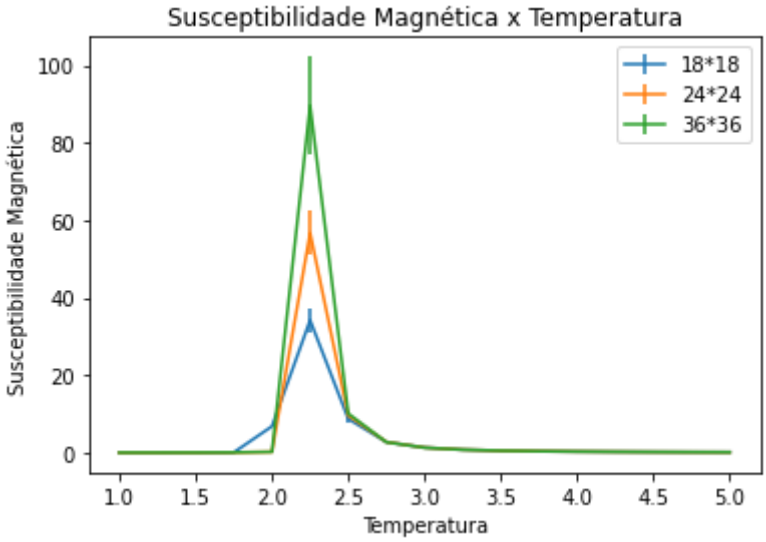
Susceptibilidade Magnética

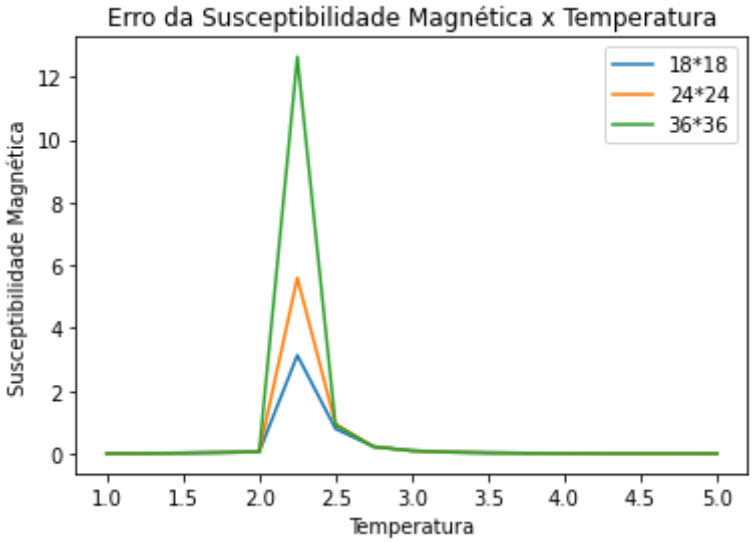
In [114…

```
# Energia
plt.title("Susceptibilidade Magnética x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Susceptibilidade Magnética")
for tam in L:
    plt.errorbar(temp, values[tam]["S"], yerr = errors[tam]["S"], label = (st
plt.legend()
plt.show()
plt.close()

# Obs: As energias foram muito semelhantes nas redes então as linhas acabaram

plt.title("Erro da Susceptibilidade Magnética x Temperatura")
plt.xlabel("Temperatura")
plt.ylabel("Susceptibilidade Magnética")
for tam in L:
    plt.plot(temp, errors[tam]["S"], label = (str(tam)+"*"+str(tam)))
plt.legend()
plt.show()
plt.close()
```





Observações:

- Espera-se que em um sistema de tamanho infinito os erros se tornem negligenciáveis e as grandezas cresçam infinitamente também, porém mantendo a proporcionalidade com a temperatura.
- É possível notar transição de fase a partir dos 3 graus, onde as grandezas tendem a ter menos variações (quase nenhuma).
- Existe um pico de erros estatísticos. Os erros tendem a acompanhar o tamanho das grandezas (Quando temos um pico nas grandezas, temos um pico nos erros e quando as grandezas aumentam os erros também tendem à aumentar.)