

## Projeto 5 - Simulações de Monte Carlo

Jeftor Santiago (12559016)

Entrega: 08/06/2024

### Introdução ao modelo de Ising

Seja uma malha 2D com  $L_x, L_y$  sítios nos eixos  $x$  e  $y$ , respectivamente. Cada sítio tem um spin  $\sigma_k = \{-1, +1\}$  associado.

$$\mathcal{H} = -\frac{J}{2} \sum_{i=1}^{L_x} \sum_{\ell=1}^{L_y} s(i, \ell) [s_{i-1, \ell} + s_{i+1, \ell} + s_{i, \ell-1} + s_{i, \ell+1}] \quad (1)$$

trabalhamos apenas com malhas regulares quadradas, então  $L \cdot L = N$ , com  $L = L_x = L_y$ .  
a energia total é

$$E = \langle \mathcal{H} \rangle \quad (2)$$

Buscamos entender a relação da energia, magnetização e a temperatura nesse sistema. Partindo da função de partição

$$\mathcal{Z}(\beta) = \sum_{i=1}^L \sum_{\ell=1}^L e^{-\beta E} \quad (3)$$

onde  $E$  é a energia dada pela (1) e  $\beta = 1/(k_B T)$ .  
Além disso, o calor específico pode ser obtido por

$$C = \frac{1}{N} \frac{\langle \mathcal{H}^2 \rangle - \langle \mathcal{H} \rangle^2}{k_B T^2} \quad (4)$$

Partindo do hamiltoniano (1) e o spin  $\sigma_k$  em cada sítio  $(i, \ell)$  podemos fazer diversas medidas como do sistema. Nesse trabalho estamos interessando principalmente na energia média e magnetização média por sítio. Sendo a magnetização de um sítio sendo o estado do seu spin, dizemos que a magnetização total é simplesmente a soma de todos spins normalizada pela quantidade total de spins  $N$ .

$$\langle m \rangle = \frac{1}{N} \sum_{i=1}^L \sum_{\ell=1}^L s(i, \ell) \quad (5)$$

Da descrição da magnetização e a eq.(5) já podemos intuir que as configurações correspondentes à máximo ou mínimo de magnetização são todos spins alinhados, sendo  $\langle m \rangle_{\max} = \pm 1$ .

Queremos estudar a relação dessas medidas de energia e magnetização e suas relações com a temperatura. Por isso buscamos uma outra relação para a magnetização média por spin. Utilizando a função de partição(3) para normalização dessa medida, temos  $Z$  constante de normalização e

$$\langle m \rangle = \frac{1}{N} \frac{1}{Z} \left( \sum_{\sigma} e^{-\beta E_{s_{\sigma}}} \right) \quad (6)$$

$$P(s) = \frac{e^{\beta s \Delta M}}{e^{-\beta s \Delta M} + e^{\beta s \Delta M}} \quad (7)$$

$$P(-s) = \frac{e^{-\beta s \Delta M}}{e^{-\beta s \Delta M} + e^{\beta s \Delta M}} \quad (8)$$

onde  $\Delta M$  é

$$\Delta M = J [s(i-1, \ell) + s(i+1, \ell) + s(i, \ell-1) + s(i, \ell+1)] \quad (9)$$

Com essa descrição da magnetização média por spin podemos impor alguma dinâmica para o sistema e observar as medidas desejadas. A dinâmica que trabalhamos nesse projeto que consiste em realizar alterar algum spin aleatório da malha de acordo com a probabilidade, isto é, sorteamos a chance de flipar o spin e se for menor ou maior (7) (8) o estado é alterado e realizamos as medidas desejadas. Utilizamos simulação de Monte Carlo para realizar a dinâmica até que o sistema atinja um determinado equilíbrio. Nesse método cada passo da simulação corresponde à dinâmica de alterar um spin para os  $N$  spins do sistema. Tipicamente nas simulações realizadas o número de passos de Monte Carlo utilizados foi 3000, mas para observar alguns fenômenos foi necessário aumentar esse número.

#### *Detalhes de implementação*

Todos os programas implementados no projeto seguem estruturas parecidas, portanto, foi criado um módulo apenas com operações básicas envolvendo o modelo de Ising em 2D. No arquivo `ising_modules.f` localizado na raiz do projeto estão as funções e rotinas gerais utilizadas nas diferentes simulações.

Na descrição da malha 2D foram adotadas condições periódicas de contorno, então as bordas da grade estão

conectadas e na discretização das posições dos spins foi utilizado um vetor de posição definido como

```
1 dimension ipbc(0:L+1)
2 N = L * L
3 ! setting periodic boundary conditions
4 do i = 1, L
5     ipbc(i) = i
6 end do
7 ipbc(0) = L
8 ipbc(L+1) = 1
```

Além disso, como as (7) (8) envolvem exponenciais e são contas feitas várias vezes no programa, é mais eficiente armazenar o valor das exponenciais em vetores para cada um dos vizinhos  $(i + 1, \ell)$ ,  $(i - 1, \ell)$ ,  $(i, \ell + 1)$ ,  $(i, \ell - 1)$  de um spin em  $(i, \ell)$ .

Algumas escolhas nas implementações podem ter sido ineficientes, como ter uma rotina apenas para inicializar o calculo da magnetização. Embora isso seja verdade, optei por fazer dessa forma para obter um código mais limpo e fácil de trabalhar. Entretanto poderia ter feito esse tipo de conta na inicialização da configuração da grade.

Segue abaixo as rotinas gerais utilizadas para medidas que envolvem o modelo de Ising:

```
1 subroutine define_exponentials(exps, beta)
2     dimension exps(-4:4)
3     do i = -4,4
4         exps(i) = exp(-beta*i)/(exp(beta*i)+exp(-beta*i))
5     end do
6 end subroutine define_exponentials
7
8 subroutine flip_spin(lattice, ipbc, exps, E, mag, L_real)
9     implicit integer(s-s, d-d)
10    implicit real(m-m)
11    parameter(L = 100)
12    parameter(J = 1.0)
13    dimension exps(-4:4)
14    byte lattice(1:L, 1:L)
15    dimension ipbc(0:L+1)
16
17    ! choose a random site
18    i = floor(rand()* L_real) + 1
19    k = floor(rand()* L_real) + 1
20
21    dM = lattice(ipbc(i-1),k) + lattice(ipbc(i+1),k)
22    dM = dM + lattice(i,ipbc(k-1)) + lattice(i,ipbc(k+1))
23    dM = J * dM
24
25    ! spin(i, k) and dM are positions of the lattice, so
26    e_flip = exps(lattice(i,k)*dM)
27    if(rand() < e_flip) then
28        lattice(i, k) = - lattice(i, k)
29    ! Magnetization
30    N = L_real*L_real
31    mag = mag + 2.0*(1.0e0*lattice(i, k)/N)
```

```

32      ! print *, "<m> = ", mag
33      ! Change in the energy
34      E = E - 2*lattice(i, k)*dM
35  end if
36 end subroutine flip_spin
37
38 function H_0(lattice, ipbc, L_real)
39   parameter(L = 100)
40   byte lattice(1:L, 1:L)
41   dimension ipbc(0:L+1)
42   H_0 = 0
43   do i = 1, L_real
44     do k = 1, L_real
45       adj = lattice(ipbc(i-1),k)+lattice(ipbc(i+1),k)
46       adj = adj+lattice(i,ipbc(k-1))+lattice(i,ipbc(k+1))
47       H_0 = H_0 + adj * lattice(i, k)
48     end do
49   end do
50   !      E = (-J/2) * (s(i, j)[s(i-1, j) + s(i+1, j) + s(i, j-1) + s(i, j+1)])
51   H_0 = - 0.5 * H_0
52 end function H_0
53
54
55 subroutine initialize_lattice(lattice, L_x, L_y)
56   implicit real(m-m)
57   parameter(L = 100)
58   byte lattice(1:L, 1:L)
59   ! initializing lattice
60   do i = 1, L_x
61     do k = 1, L_y
62       lattice(i, k) = 1
63     end do
64   end do
65 end subroutine initialize_lattice
66
67 subroutine initialize_random_lattice(lattice, L_x, L_y)
68   implicit real(m-m)
69   parameter(L = 100)
70   byte lattice(1:L, 1:L)
71   ! initializing lattice
72   do i = 1, L_x
73     do k = 1, L_y
74       if(rand() < 0.5) then
75         lattice(i, k) = 1
76       else
77         lattice(i, k) = -1
78       end if
79     end do
80   end do
81 end subroutine initialize_random_lattice
82
83 subroutine total_magnetization(lattice, mag, L_real)
84   implicit real(m-m)
85   parameter(L = 100)
86   byte lattice(1:L, 1:L)
87   N = L_real * L_real
88   mag = 0.0e0
89   do i = 1, L_real
90     do j = 1, L_real
91       mag = mag + 1.0e0 * lattice(i, j)
92     end do

```

```

93     end do
94     mag = mag / N
95 end subroutine total_magnetization
96
97 subroutine write_lattice(lattice, L_real, f_name)
98     implicit integer (f-f)
99     parameter(L = 100)
100    byte lattice(1:L, 1:L)
101    character *1 symb(-1:1)
102    symb(1) = '1'
103    symb(-1) = '0'
104    do i = 1, L_real
105        write(f_name, '(100A2)') (symb(lattice(i,j)), j = 1, L_real)
106    end do
107 end subroutine write_lattice

```

## Tarefa A - Dinâmica de Monte Carlo para temperaturas fixas

Nessa tarefa foi simulado a dinâmica de Monte Carlo para sistemas com temperaturas iniciais relativas à  $\beta = 3$  e  $\beta = 0.1$ . Para temperaturas altas,  $\beta$  pequeno, é esperado que o sistema não atinja o equilíbrio após a dinâmica. O oposto acontece com a dinâmica de temperatura baixa, o sistema fica ordenado e a magnetização ficará nula.

As simulação foram feitas para malhas de tamanhos  $L = 60$  e  $L = 100$ . Segue abaixo a estrutura a implementação das simulações:

```

1
2 open(1, file="saidas/tarefa-1/saida-tarefa-A1-conf-L60.dat")
3 open(3, file="saidas/tarefa-1/saida-tarefa-A1-conf-L100.dat")
4 open(5, file="saidas/tarefa-1/saida-tarefa-A2-conf-L60.dat")
5 open(7, file="saidas/tarefa-1/saida-tarefa-A2-conf-L100.dat")
6
7 open(2, file="saidas/tarefa-1/saida-tarefa-A1-energia-L60.dat")
8 open(4, file="saidas/tarefa-1/saida-tarefa-A1-energia-L100.dat")
9 open(6, file="saidas/tarefa-1/saida-tarefa-A2-energia-L60.dat")
10 open(8, file="saidas/tarefa-1/saida-tarefa-A2-energia-L100.dat")
11
12 call tarefal(60, 3.0, 1, 2)
13 call tarefal(60, 0.1, 5, 4)
14
15 call tarefal(100, 3.0, 3, 6)
16 call tarefal(100, 0.1, 7, 8)
17
18 do i = 1, 8
19     close(i)
20 end do
21 end
22
23 subroutine tarefal(L_real, beta, fname1, fname2)
24     implicit integer(f-f)
25     implicit real(m-m)
26     parameter(L = 100)
27
28     dimension exps(-4:4)

```

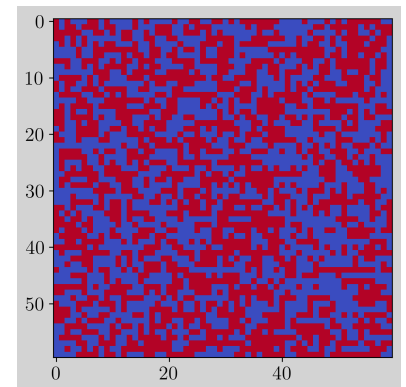


Figura 1: Configuração final para malha de tamanho  $L = 100$ .

```

29  byte lattice(1:L, 1:L)
30  ! periodic boundary conditions
31  dimension ipbc(0:L+1)
32  ! this or using mod
33
34  N = L_real * L_real
35  ! setting ipbc
36  do i = 1, L_real
37      ipbc(i) = i
38  end do
39  ipbc(0) = L_real
40  ipbc(L_real+1) = 1
41
42  m = 0
43
44  call define_exponentials(exps, beta)
45
46  call initialize_lattice(lattice, L_real, L_real)
47
48  ! initial energy
49  E = H_0(lattice, ipbc, L_real)
50
51  write(fname2, *) 0, E
52  call srand(iseed)
53  ! initialize monte carlo dynamics
54  do k = 1, 3000
55      ! sweeps all configurations
56      ! randomly flips spins
57      do i = 1, N
58          call flip_spin(lattice, ipbc, exps, E, m, L_real)
59      end do
60      write(fname2, *) k, E / N
61  end do
62  call write_lattice(lattice, L_real, fname1)
63  end subroutine tarefal

```

A.1 -  $\beta = 3$

Podemos constatar pelas (3) e (1) que o sistema fica totalmente ordenado. A magnetização do sistema é  $\langle m \rangle = \pm 1$  dependendo da orientação da ordenação.

A.2 -  $\beta = 0.1$

Para temperaturas mais altas o sistema fica desordenado, sem uma orientação fixa para todos spins da malha, nesse caso a simetria vai existir magnetização mas essa é balanceada pelas orientações contrárias e se cancela.

Pelas figuras (??) e (??) podemos observar o estado final do sistema para diferentes tamanhos de grade.

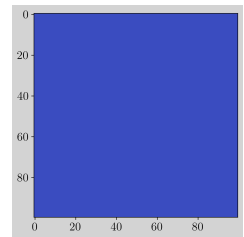


Figura 2: Configuração final para malha de tamanho L=60.

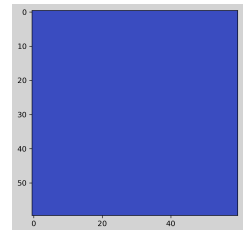


Figura 3: Configuração final para malha de tamanho  $L = 60$ .

## Tarefa B - Processos de recozimento e têmpera

### B.1 - Recozimento

O programa desenvolvido para essa simulação está abaixo:

```
1      !      Tarefa B - Recozimento e quenching
2      implicit real(j-j, m-m)
3      parameter(L = 100)
4      dimension exps(-4:4)
5      byte lattice(1:L, 1:L)
6
7      ! periodic boundary conditions
8      dimension ipbc(0:L+1)
9
10     L_real = 90
11
12     do i = 1, L_real
13         ipbc(i) = i
14     end do
15     ipbc(0) = L_real
16     ipbc(L_real+1) = 1
17
18     N = L_real * L_real
19
20     mag = 0.0d0
21
22     call srand(351324)
23
24     call initialize_random_lattice(lattice, L_real, L_real)
25
26     open(1, file="saidas/tarefa-2/saida-tarefa-B1-conf-inicial.dat")
27     open(2, file="saidas/tarefa-2/saida-tarefa-B1-conf-final.dat")
28     open(3, file="saidas/tarefa-2/saida-tarefa-B1-mag-eng.dat")
29
30     call write_lattice(lattice, L_real, 1)
31     call total_magnetization(lattice, mag, L_real)
32
33     ! initial energy
34     E = H_0(lattice, ipbc, L_real)
35
36     dbeta = 0.001
37     ! monte carlo dynamics
38     do i = 1, 3000
39         beta = i * dbeta
40         call define_exponentials(exps, beta)
41         do k = 1, N
42             call flip_spin(lattice, ipbc, exps, E, mag, L_real)
43         end do
44         write(3, *) i, mag, E/N
45     end do
46     call write_lattice(lattice, L_real, 2)
47     close(1)
48     close(2)
49     close(3)
50     end
```

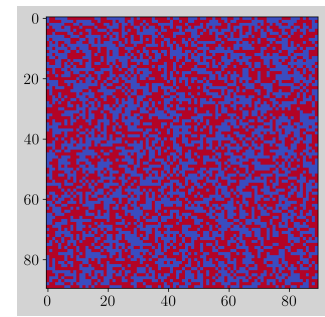


Figura 5: Configuração inicial da simulação.  $\beta = 1/2$

Fazendo evolução da temperatura de forma lenta, com  $\Delta\beta = 0.001$  temos o processo de recozimento. Partimo do sistema desordenado, com temperatura infinita e a cada

passo de Monte Carlo provocamos uma variação de temperatura  $\Delta\beta$ . A figura (5) mostra a configuração inicial do sistema de spins.

Estamos interessados em observar a energia média por spin. Pelo gráfico abaixo(6) nota-se que a energia média parte de zero, pois o sistema está completamente desordenado, e decresce até atingir a energia limite em  $-2$ .

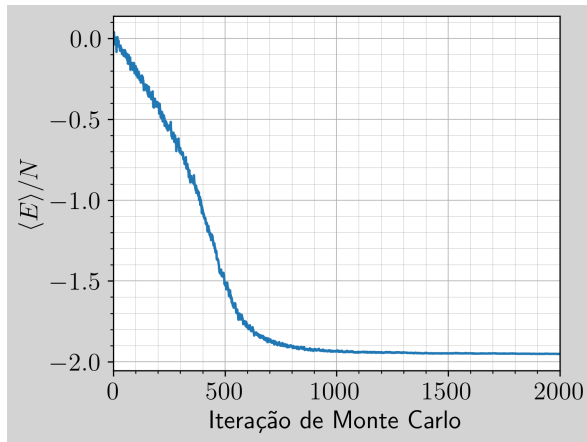


Figura 6: Energia média de spin por iterações de Monte Carlo.

Além disso, temos a configuração final dos spins do sistema bidimensional(7). Há uma faixa de magnetização na malha, a presença dela deve estar associada ao número de iterações de Monte Carlo feita utilizado na simulação (3000 passos) que não foi o bastante para o sistema ficar todo alinhado.

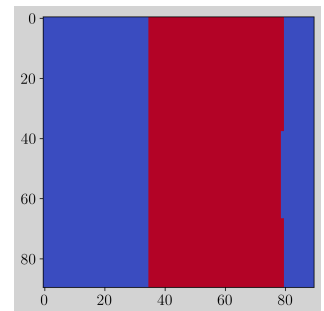


Figura 7: Configuração final da malha 2D após dinâmica de recozimento.

## B.2 - Tempera

O código dessa simulação é quase idêntico ao da tarefa anterior e está compilado abaixo:

```
1  implicit real(j-j, m-m)
2  parameter(L = 100)
3  dimension exps(-4:4)
4  byte lattice(1:L, 1:L)
5
6  ! periodic boundary conditions
7  dimension ipbc(0:L+1)
8
9  L_real = 90
10
11 do i = 1, L_real
12     ipbc(i) = i
13 end do
14
15 ipbc(0) = L_real
16 ipbc(L_real+1) = 1
17
18 N = L_real * L_real
19
20 mag = 0.0d0
21
```

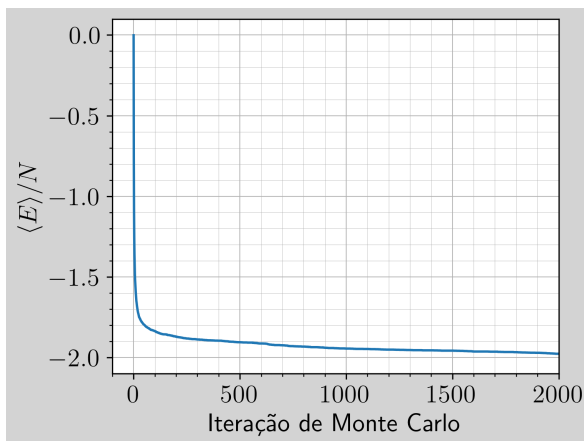


```

22  call srand(96312)
23  call initialize_random_lattice(lattice, L_real, L_real)
24
25  open(1, file="saidas/tarefa-2/saida-tarefa-B2-conf-inicial.dat")
26  open(2, file="saidas/tarefa-2/saida-tarefa-B2-conf-final.dat")
27  open(3, file="saidas/tarefa-2/saida-tarefa-B2-mag-eng.dat")
28
29  call write_lattice(lattice, L_real, 1)
30
31  call total_magnetization(lattice, mag, L_real)
32
33  ! initial energy
34  E = H_0(lattice, ipbc, L_real)
35
36
37  dbeta = 0.001
38  write(3, *) 0, mag, E/N
39  ! monte carlo dynamics
40  beta = 3
41
42  call define_exponentials(exps, beta)
43  do i = 1, 3000
44      do k = 1, N
45          call flip_spin(lattice, ipbc, exps, E, mag, L_real)
46      end do
47      write(3, *) i, mag, E/N
48  end do
49  call write_lattice(lattice, L_real, 2)
50  close(1)
51  close(2)
52  close(3)
53  end

```

Nessa simulação partimos da mesma configuração inicial que a anterior e variamos o  $\beta$  de maneira brusca e o sistema pode não atingir o equilíbrio. Foi utilizada uma malha de tamanho  $L = 90$  para essa simulação e mesmo número de passos.



Nota-se que a energia média decaí muito mais rapidamente que no caso anterior(6) e a configuração final consegue atingir o equilíbrio(9). Diferentemente do processo anterior, na temperatura os spins vizinhos conseguem se alinhar no tempo de Monte Carlo utilizado na simulação.

Figura 8: Energia média de spin por iterações de Monte Carlo.

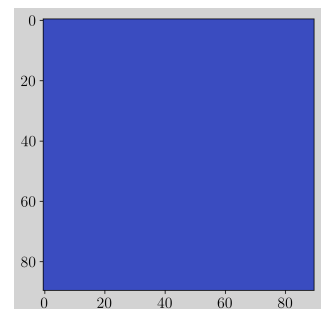


Figura 9: Configuração final para rede de spins na dinâmica de temperatura.

## Tarefa C - Loop térmico

### C.1 - Histerese

Segue abaixo a implementação da simulação de histerese:

```
1  open(1, file="saidas/tarefa-3/saida-tarefa-C1-L60-DB1.dat")
2  open(2, file="saidas/tarefa-3/saida-tarefa-C1-L60-DB2.dat")
3
4  open(3, file="saidas/tarefa-3/saida-tarefa-C1-L80-DB1.dat")
5  open(4, file="saidas/tarefa-3/saida-tarefa-C1-L80-DB2.dat")
6
7  open(5, file="saidas/tarefa-3/saida-tarefa-C1-L100-DB1.dat")
8  open(6, file="saidas/tarefa-3/saida-tarefa-C1-L100-DB2.dat")
9
10
11  call tarefaC1(60, 0.001, 1)
12  call tarefaC1(60, 0.0001, 2)
13
14  call tarefaC1(80, 0.001, 3)
15  call tarefaC1(80, 0.0001, 4)
16
17  call tarefaC1(100,0.001, 5)
18  call tarefaC1(100,0.0001, 6)
19
20  do i = 1, 6
21      close(i)
22  end do
23  end
24
25  subroutine tarefaC1(L_real, dbeta, f_name)
26      implicit integer(f-f)
27      !          Tarefa B - Recozimento e quenching
28      implicit real(j-j, m-m)
29      parameter(L = 100)
30      dimension exps(-4:4)
31      byte lattice(1:L, 1:L)
32
33      ! periodic boundary conditions
34      dimension ipbc(0:L+1)
35
36      do i = 1, L_real
37          ipbc(i) = i
38      end do
39
40      ipbc(0) = L_real
41      ipbc(L_real+1) = 1
42
43      N = L_real * L_real
44
45      mag = 0.0d0
46
47      call srand(96312)
48      ! b = 0
49      call initialize_random_lattice(lattice, L_real, L_real)
50
51      call total_magnetization(lattice, mag, L_real)
52
53      ! initial energy
```

```

54     E = H_0(lattice, ipbc, L_real)
55
56     beta = 0.0
57
58     write(f_name, *) 0, beta, mag, E/N
59
60     imax = int(1.75 / dbeta) + 1
61
62     do i = 1, imax
63
64         call define_exponentials(exps, beta)
65
66         if(i < imax/ 2) then
67             beta = beta + dbeta
68         else
69             beta = beta - dbeta
70         end if
71
72         do k = 1 , N
73             call flip_spin(lattice,ipbc,exps,E,mag,L_real)
74         end do
75
76         write(f_name, *) i, beta, mag, E/N
77
78     end do
79 end subroutine tarefaC1

```

No gráfico (10) temos o comportamento da energia média por spin na dinâmica do loop térmico e o gráfico de histerese, isto é, a energia média em relação à  $\beta$  para variações de  $\Delta b = 0,001$ .

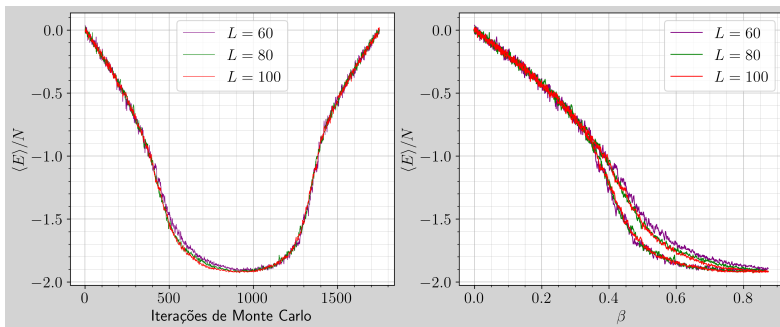


Figura 10: À esquerda energia média por spin por iterações de Monte Carlo e à direita em relação à  $\beta$ .

A figura (11) equivale a dinâmica como a anterior, mas com uma variação  $\Delta\beta = 0,0001$ , que fornece um resultado com menos flutuações, sobretudo para as redes maiores.

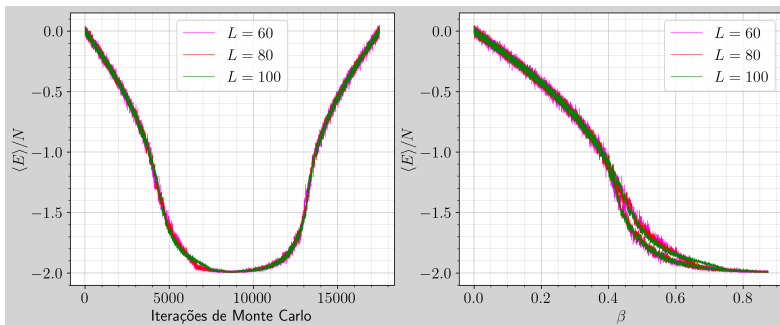


Figura 11: À esquerda energia média por spin por iterações de Monte Carlo e à direita em relação à  $\beta$ .

Podemos observar nos gráficos que a região de histerese

correspondem à um intervalo de  $\beta$  entre 0,4 e 0,6, mas apenas a partir dessas medidas não conseguimos ter uma boa precisão dessa medida.

## C.2 - Temperatura crítica

Modificação no código do item anterior:

```

1  dimension betas(1:5)
2  parameter(betas = (/0.41, 0.44, 0.47, 0.51, 0.55/))
3
4
5  open(1, file="saidas/tarefa-3/saida-tarefa-C2-L60-b1.dat")
6  open(2, file="saidas/tarefa-3/saida-tarefa-C2-L60-b2.dat")
7  open(3, file="saidas/tarefa-3/saida-tarefa-C2-L60-b3.dat")
8  open(4, file="saidas/tarefa-3/saida-tarefa-C2-L60-b4.dat")
9  open(5, file="saidas/tarefa-3/saida-tarefa-C2-L60-b5.dat")
10
11 do i = 1, 5
12     call tarefaC2(60, betas(i), i)
13     close(1)
14 end do
15
16 open(1, file="saidas/tarefa-3/saida-tarefa-C2-L80-b1.dat")
17 open(2, file="saidas/tarefa-3/saida-tarefa-C2-L80-b2.dat")
18 open(3, file="saidas/tarefa-3/saida-tarefa-C2-L80-b3.dat")
19 open(4, file="saidas/tarefa-3/saida-tarefa-C2-L80-b4.dat")
20 open(5, file="saidas/tarefa-3/saida-tarefa-C2-L80-b5.dat")
21
22 do i = 1, 5
23     call tarefaC2(80, betas(i), i)
24     close(1)
25 end do
26
27 open(1, file="saidas/tarefa-3/saida-tarefa-C2-L100-b1.dat")
28 open(2, file="saidas/tarefa-3/saida-tarefa-C2-L100-b2.dat")
29 open(3, file="saidas/tarefa-3/saida-tarefa-C2-L100-b3.dat")
30 open(4, file="saidas/tarefa-3/saida-tarefa-C2-L100-b4.dat")
31 open(5, file="saidas/tarefa-3/saida-tarefa-C2-L100-b5.dat")
32
33 do i = 1, 5
34     call tarefaC2(100, betas(i), i)
35     close(1)
36 end do
37 end
38 subroutine tarefaC2(L_real, beta, fname)
39     ! Tarefa B - Recozimento e quenching
40     implicit integer(f-f)
41     implicit real(j-j, m-m)
42     parameter(L = 100)
43     dimension exps(-4:4)
44     byte lattice(1:L, 1:L)
45     ! periodic boundary conditions
46     dimension ipbc(0:L+1)
47
48     do i = 1, L_real
49         ipbc(i) = i
50     end do
51
52     ipbc(0) = L_real

```

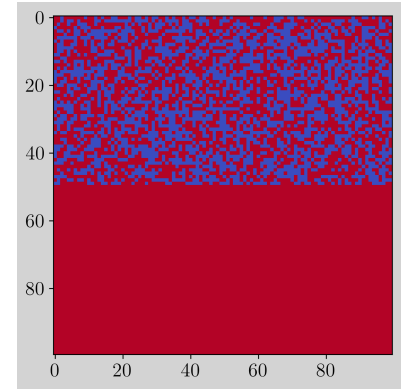


Figura 12: Configuração inicial para dinâmica utilizada na medida de temperatura crítica.

```

53     ipbc(L_real+1) = 1
54
55     N = L_real * L_real
56
57     mag = 0.0d0
58
59     call srand(L_real * 392)
60
61     ! half ordered / half random.
62     call initialize_lattice(lattice, L_real, L_real)
63     call initialize_random_lattice(lattice, L_real/2, L_real)
64
65     open(99, file = "saidas/tarefa-3/saida-tarefa-C2-conf.dat")
66     call write_lattice(lattice, L_real, 99)
67     close(99)
68
69     call total_magnetization(lattice, mag, L_real)
70
71     ! initial energy
72     E = H_0(lattice, ipbc, L_real)
73     dbeta = 0.01
74     write(fname, *) 0, E/N
75     do i = 1, 3000
76         call define_exponentials(exps, beta)
77         do k = 1, N
78             call flip_spin(lattice, ipbc, exps, E, mag, L_real)
79         end do
80         write(fname, *) i, E/N
81     end do
82 end subroutine tarefaC2

```

Partimos dos resultados do item anterior e tentamos obter a temperatura crítica do modelo. Para isso observamos a variação de energia no intervalo  $\beta$  discutido antes, isto é,  $0,4 < \beta < 0,6$ . A imagem (12) mostra a configuração inicial do sistema. Foram escolhidos alguns valores de  $\beta$  para executar a dinâmica de Monte Carlo.

Nas figuras (14), (13) e (15) estão as evoluções, em um intervalo de passos de Monte Carlo reduzido, da energia média por spin.

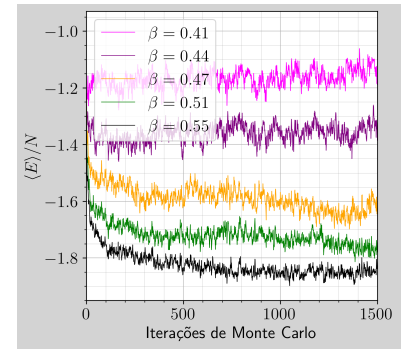


Figura 13: Dinâmica para  $L=80$ .

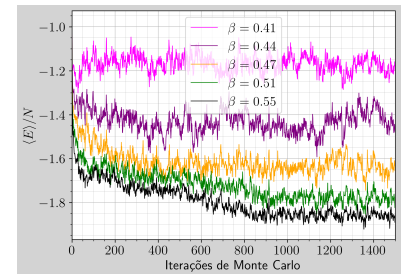


Figura 14: Dinâmica para  $L=60$ .

Nota-se que as energias médias por spin sempre partem do mesmo valor no intervalo da histerese e a que possui maior variação é a que corresponde à  $\beta = 0.44$ , esse é o  $\beta$  relacionado à temperatura crítica  $T_c = 1/\beta_c \approx 2,27$ .

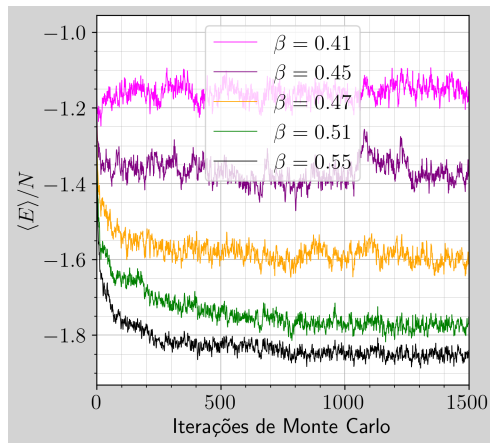


Figura 15: Dinâmica para  $L=100$ .

Além disso, pela (4) podemos constatar que esse  $\beta_c = 0,44$  também está associado à um valor específico crítico no modelo.

#### Tarefa D - Quebra espontânea de simetria

Nessa tarefa o nosso interesse é estudar o fenômeno de quebra espontânea de simetria. Queremos mostrar que o tempo que um sistema leva para mudar toda a orientação de magnetização cresce de forma exponencial com a dimensão da malha utilizada. Para isso foi implementado uma simulação que executa passos de Monte Carlo e contabiliza o intervalo de tempo de Monte Carlo que o sistema leva para mudar a magnetização conforme o tamanho  $L$  da rede aumenta.

O código em fortran para essa simulação está abaixo:

```

1  implicit integer(f-f)
2  implicit real(m-m)
3  parameter(L = 100)
4  dimension exps(-4:4)
5  byte lattice(1:L, 1:L)
6  ! periodic boundary conditions
7  dimension ipbc(0:L+1)
8  ! this or using mod
9  open(unit=1, file="saidas/tarefa-4/saida-tarefa-D.dat")
10 open(unit=2, file="saidas/tarefa-4/saida-tarefa-MAG.T.dat")
11 beta = 0.5
12 call define_exponentials(exps, beta)
13 do L_real = 4, 10
14   print *, "L = ", L_real
15   call srand(3519) ! /L_real+1
16   N = L_real * L_real
17   ! setting ipbc
18   do i = 1, L_real

```

```

19     ipbc(i) = i
20 end do
21 ipbc(0) = L_real
22 ipbc(L_real+1) = 1
23 mag = 0.0e0
24 call initialize_random_lattice(lattice, L_real, L_real)
25 call total_magnetization(lattice, mag, L_real)
26 n_inversions = 10000
27 n_curr = 0
28 n_time = 0
29 do while(n_curr < n_inversions)
30     mag_prev = mag
31     do i = 1, N
32         call flip_spin(lattice, ipbc, exps, E, mag, L_real)
33     end do
34     n_time = n_time + 1
35     ! Fazer o gráfico da magnetização aqui.
36     if(mag_prev * mag < 0) then
37         t_mean = t_mean + n_time
38         n_time = 0
39         n_curr = n_curr + 1
40     end if
41 end do
42 write(2, *) t_mean, mag
43 t_mean = t_mean / n_inversions
44 write(1, *) L_real, t_mean
45 end do
46 close(1)
47 end

```

Podemos ver pela figura(16) que o intervalo cresce de forma exponencial com o tamanho  $L$  da malha:

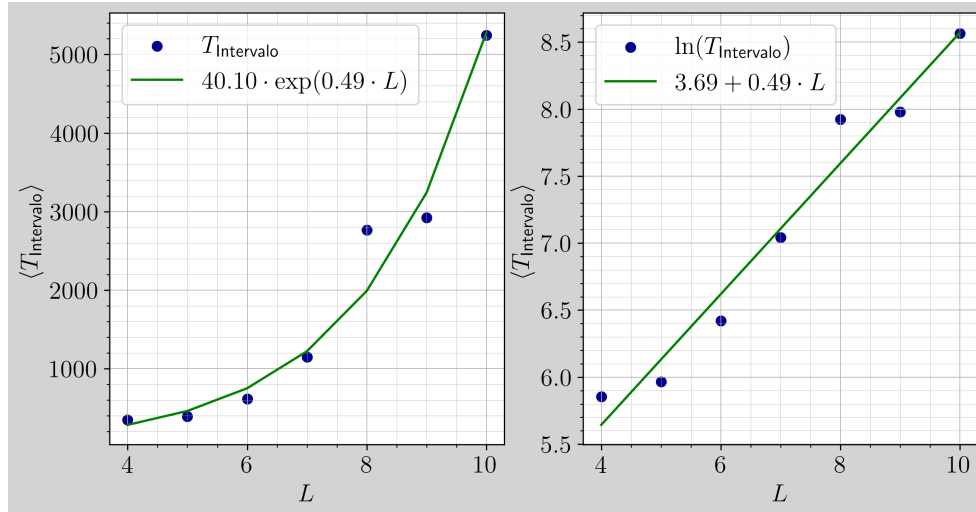


Figura 16: Gráfico do crescimento do intervalo  $\langle T_{\text{intervalo}} \rangle$  em função de  $L$  e ajuste linear.

Para implementação com número de inversões da ordem de  $10^4$  foi obtido o ajuste linear  $\ln(\langle T_{\text{intervalo}}(L) \rangle) \approx 3,67 + 0,49 \cdot L$ .

Essa dependência exponencial para que ocorra quebra da simetria talvez explique o que ocorre na simulação da tarefa B em que o sistema atinge o equilíbrio mas a magnetização tem comportamento não usual. Aumentando

o número de passos de Monte Carlo naquela simulação pode resolver o aparente problema.