Projeto 5 - Simulações de Monte Carlo

Jefter 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.

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

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 (**??**) 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 (**??**) e o spin σ_k em cada sítio (i, ℓ) podemos fazer diversas medidas como do sistema. Nesse trabalho estamos interessandos principalemente 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.(**??**) 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(**??**) para normalização dessa medida, temos Z constante de normalização e

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

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

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

onde ΔM é

$$\Delta M = J\left[s(i-1,\ell) + s(i+1,\ell) + s(i,\ell-1) + s(i,\ell+1)\right] \tag{9}$$

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 (**??**) (**??**) 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

```
    dimension ipbc(0:L+1)
    N = L * L
    ! setting periodic boundary conditions
    do i = 1, L
        ipbc(i) = i
    end do
    ipbc(0) = L
    ipbc(L+1) = 1
```

Além disso, como as (**??**) (**??**) 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,ℓ).

Algumas escolhas nas implementações podem ter sido ineficientes, como ter uma rotina apenas para initializar o calculo da magnetização. Embora isso seja verdade, optei por fazer dessa forma para obter um código mais limpo e fácil de trabalhar. Entretando 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:

```
    subroutine define_exponentials(exps, beta)
      dimension exps(-4:4)
      do i = -4,4
          exps(i) = exp(-beta*i)/(exp(beta*i)+exp(-beta*i))
      end do
    end subroutine define_exponentials

    subroutine flip_spin(lattice, ipbc, exps, E,  mag, L_real)
      implicit integer(s-s, d-d)
      implicit real(m-m)
      parameter(L = 100)
      parameter(J = 1.0)
      dimension exps(-4:4)
      byte lattice(1:L, 1:L)
      dimension ipbc(0:L+1)

      ! choose a random site
      i = floor(rand()* L_real) + 1
      k = floor(rand()* L_real) + 1

      dM = lattice(ipbc(i-1),k) + lattice(ipbc(i+1),k)
      dM = dM + lattice(i,ipbc(k-1)) + lattice(i,ipbc(k+1))
      dM = J * dM

      ! spin(i, k) and dM are positions of the lattice, so
      e_flip =  exps(lattice(i,k)*dM)
      if(rand() < e_flip) then
          lattice(i, k) = - lattice(i, k)
          ! Magnetization
          N = L_real*L_real
          mag = mag + 2.0*(1.0e0*lattice(i, k)/N)
```

```
        ! print *, "<m> = ", mag
        ! Change in the energy
        E = E - 2*lattice(i, k)*dM
      end if
    end subroutine flip_spin

    function H_0(lattice, ipbc, L_real)
      parameter(L = 100)
      byte lattice(1:L, 1:L)
      dimension ipbc(0:L+1)
      H_0 = 0
      do i = 1, L_real
         do k = 1, L_real
             adj = lattice(ipbc(i-1),k)+lattice(ipbc(i+1),k)
             adj = adj+lattice(i,ipbc(k-1))+lattice(i,ipbc(k+1))
             H_0 = H_0 + adj * lattice(i, k)
         end do
      end do
    !           E = (-J/2) * (s(i, j)[s(i  1, j) + s(i + 1, j) + s(i, j  1) + s(i, j + 1))
      H_0 = - 0.5 * H_0
    end function H_0


    subroutine initialize_lattice(lattice, L_x, L_y)
      implicit real(m-m)
      parameter(L = 100)
      byte lattice(1:L, 1:L)
      ! initializing lattice
      do i = 1, L_x
         do k = 1, L_y
            lattice(i, k) = 1
         end do
      end do
    end subroutine initialize_lattice

    subroutine initialize_random_lattice(lattice, L_x, L_y)
      implicit real(m-m)
      parameter(L = 100)
      byte lattice(1:L, 1:L)
      ! initializing lattice
      do i = 1, L_x
          do k = 1, L_y
             if(rand() < 0.5) then
                 lattice(i, k) = 1
              else
                  lattice(i, k) = -1
              end if
          end do
      end do
    end subroutine initialize_random_lattice

    subroutine total_magnetization(lattice, mag, L_real)
      implicit real(m-m)
      parameter(L = 100)
      byte lattice(1:L, 1:L)
      N = L_real * L_real
      mag = 0.0e0
      do i = 1, L_real
          do j = 1, L_real
              mag = mag + 1.0e0 * lattice(i, j)
          end do
```

```
      end do
      mag = mag / N
    end subroutine total_magnetization

    subroutine write_lattice(lattice, L_real, f_name)
      implicit integer (f-f)
      parameter(L = 100)
      byte lattice(1:L, 1:L)
      character *1 symb(-1:1)
      symb(1) = '1'
      symb(-1) = '0'
      do i = 1, L_real
         write(f_name,'(100A2)') (symb(lattice(i,j)),j = 1,L_real)
      end do
    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, β pequeno, é esperado que o sistema não atinja o equílibrio 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:

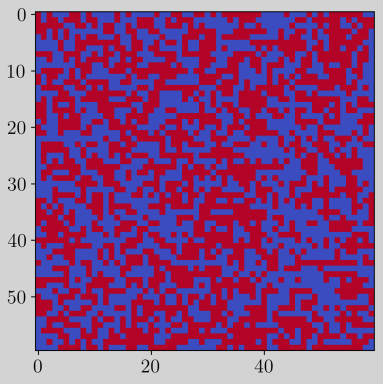

Figura 1: Configuração final para malha de tamanho $L = 100$.

```

    open(1, file="saidas/tarefa-1/saida-tarefa-A1-conf-L60.dat")
    open(3, file="saidas/tarefa-1/saida-tarefa-A1-conf-L100.dat")
    open(5, file="saidas/tarefa-1/saida-tarefa-A2-conf-L60.dat")
    open(7, file="saidas/tarefa-1/saida-tarefa-A2-conf-L100.dat")

    open(2, file="saidas/tarefa-1/saida-tarefa-A1-energia-L60.dat")
    open(4, file="saidas/tarefa-1/saida-tarefa-A1-energia-L100.dat")
    open(6, file="saidas/tarefa-1/saida-tarefa-A2-energia-L60.dat")
    open(8, file="saidas/tarefa-1/saida-tarefa-A2-energia-L100.dat")

    call tarefa1(60, 3.0, 1, 2)
    call tarefa1(60, 0.1, 5, 4)

    call tarefa1(100, 3.0, 3, 6)
    call tarefa1(100, 0.1, 7, 8)

    do i = 1, 8
        close(i)
    end do
    end

    subroutine tarefa1(L_real, beta, fname1, fname2)
        implicit integer(f-f)
        implicit real(m-m)
        parameter(L = 100)

        dimension exps(-4:4)
```

```
        byte lattice(1:L, 1:L)
        ! periodic boundary conditions
        dimension ipbc(0:L+1)
        ! this or using mod

        N = L_real * L_real
        ! setting ipbc
        do i = 1, L_real
            ipbc(i) = i
        end do
        ipbc(0) = L_real
        ipbc(L_real+1) = 1

        m = 0

        call define_exponentials(exps, beta)

        call initialize_lattice(lattice, L_real, L_real)

         ! initial energy
        E = H_0(lattice, ipbc, L_real)

        write(fname2, *) 0, E
        call srand(iseed)
         ! intialize monte carlo dynamics
        do k = 1, 3000
            ! sweeps all configurations
            ! randomly flips spins
            do i = 1 , N
               call flip_spin(lattice,ipbc,exps,E,m,L_real)
            end do
            write(fname2, *) k, E / N
        end do
        call write_lattice(lattice, L_real, fname1)
    end subroutine tarefa1
```

A.1 - $\beta = 3$

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

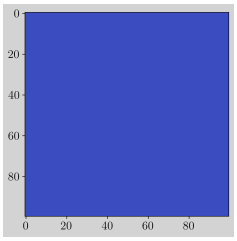

Figura 2: Configuração final para malha de tamanho L=60.

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.

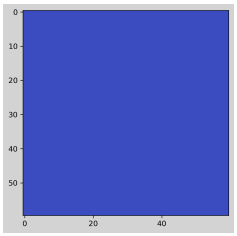

Figura 3: Configuração final para malha de tamanho $L = 60$.

Tarefa B - Processos de recozimento e têmpera

B.1 - Recozimento

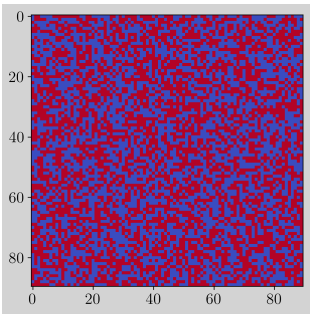

Figura 5: Configuração inicial da simulação. $\beta = 1/2$

O programa desenvolvido para essa simulação está abaixo:

```
!       Tarefa B - Recozimento e quenching
implicit real(j-j, m-m)
parameter(L = 100)
dimension exps(-4:4)
byte lattice(1:L, 1:L)

! periodic boundary conditions
dimension ipbc(0:L+1)

L_real = 90

do i = 1, L_real
    ipbc(i) = i
end do
ipbc(0) = L_real
ipbc(L_real+1) = 1

N = L_real * L_real

mag = 0.0d0

call srand(351324)

call initialize_random_lattice(lattice,  L_real, L_real)

open(1, file="saidas/tarefa-2/saida-tarefa-B1-conf-inicial.dat")
open(2, file="saidas/tarefa-2/saida-tarefa-B1-conf-final.dat")
open(3, file="saidas/tarefa-2/saida-tarefa-B1-mag-eng.dat")

call write_lattice(lattice, L_real, 1)
call total_magnetization(lattice, mag, L_real)

! initial energy
E = H_0(lattice, ipbc, L_real)

dbeta = 0.001
! monte carlo dynamics
do i = 1, 3000
    beta = i * dbeta
    call define_exponentials(exps, beta)
    do k = 1 , N
          call flip_spin(lattice,ipbc,exps,E,mag,L_real)
    end do
    write(3, *) i, mag, E/N
end do
call write_lattice(lattice, L_real, 2)
close(1)
close(2)
close(3)
end
```

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 (**??**) mostra a configuração inicial do sistema de spins.

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

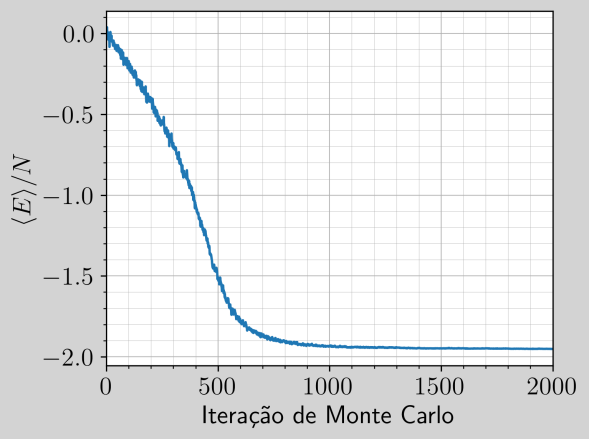

Figura 6: Energia média de spin por iterações de Monte Carlo.

Além disso, temos a configuração final dos spins do sistema bidimensional(**??**). 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.

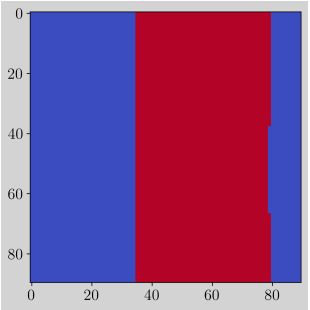

Figura 7: Configuração final da malha 2D após dinâmica de recozimento.

B.2 - Tempera

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

```
        implicit real(j-j, m-m)
        parameter(L = 100)
        dimension exps(-4:4)
        byte lattice(1:L, 1:L)

        ! periodic boundary conditions
        dimension ipbc(0:L+1)

        L_real = 90

        do i = 1, L_real
            ipbc(i) = i
        end do

        ipbc(0) = L_real
        ipbc(L_real+1) = 1

        N = L_real * L_real

        mag = 0.0d0

```

```
        call srand(96312)
        call initialize_random_lattice(lattice,  L_real, L_real)

        open(1, file="saidas/tarefa-2/saida-tarefa-B2-conf-inicial.dat")
        open(2, file="saidas/tarefa-2/saida-tarefa-B2-conf-final.dat")
        open(3, file="saidas/tarefa-2/saida-tarefa-B2-mag-eng.dat")

        call write_lattice(lattice, L_real, 1)

        call total_magnetization(lattice, mag, L_real)

        ! initial energy
        E = H_0(lattice, ipbc, L_real)


        dbeta = 0.001
        write(3, *) 0, mag, E/N
        ! monte carlo dynamics
        beta = 3

        call define_exponentials(exps, beta)
        do i = 1, 3000
            do k = 1 , N
                  call flip_spin(lattice,ipbc,exps,E,mag,L_real)
            end do
            write(3, *) i, mag, E/N
        end do
        call write_lattice(lattice, L_real, 2)
        close(1)
        close(2)
        close(3)
        end
```

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

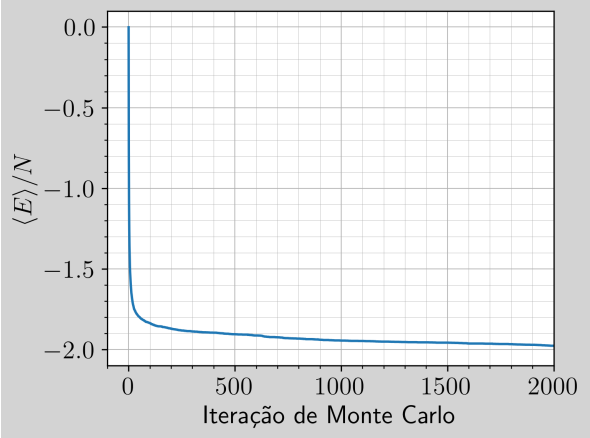

Figura 8: Energia média de spin por iterações de Monte Carlo.

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

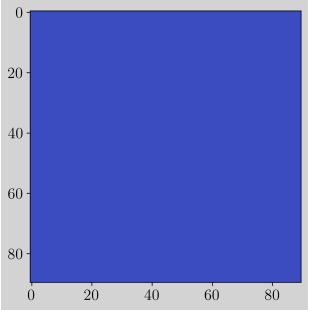

Figura 9: Configuração final para rede de spins na dinâmica de têmpera.

Tarefa C - Loop térmico

C.1 - Histerese

Segue abaixo a implementação da simulção de histerese:

```
            open(1, file="saidas/tarefa-3/saida-tarefa-C1-L60-DB1.dat")
            open(2, file="saidas/tarefa-3/saida-tarefa-C1-L60-DB2.dat")

            open(3, file="saidas/tarefa-3/saida-tarefa-C1-L80-DB1.dat")
            open(4, file="saidas/tarefa-3/saida-tarefa-C1-L80-DB2.dat")

            open(5, file="saidas/tarefa-3/saida-tarefa-C1-L100-DB1.dat")
            open(6, file="saidas/tarefa-3/saida-tarefa-C1-L100-DB2.dat")


            call tarefaC1(60, 0.001, 1)
            call tarefaC1(60, 0.0001,  2)

            call tarefaC1(80, 0.001, 3)
            call tarefaC1(80, 0.0001, 4)

            call tarefaC1(100,0.001, 5)
            call tarefaC1(100,0.0001, 6)

            do i = 1, 6
                close(i)
            end do
            end

            subroutine tarefaC1(L_real, dbeta, f_name)
                implicit integer(f-f)
            !           Tarefa B - Recozimento e quenching
                implicit real(j-j, m-m)
                parameter(L = 100)
                dimension exps(-4:4)
                byte lattice(1:L, 1:L)

                ! periodic boundary conditions
                dimension ipbc(0:L+1)

                do i = 1, L_real
                    ipbc(i) = i
                end do

                ipbc(0) = L_real
                ipbc(L_real+1) = 1

                N = L_real * L_real

                mag = 0.0d0

                call srand(96312)
                ! b = 0
                call initialize_random_lattice(lattice,  L_real, L_real)

                call total_magnetization(lattice, mag, L_real)

                ! initial energy
```

```
                E = H_0(lattice, ipbc, L_real)

                beta = 0.0

                write(f_name, *) 0, beta,  mag, E/N

                imax = int(1.75 / dbeta) + 1

                do i = 1, imax

                    call define_exponentials(exps, beta)

                    if(i < imax/ 2) then
                        beta = beta + dbeta
                    else
                        beta = beta - dbeta
                    end if

                    do k = 1 , N
                        call flip_spin(lattice,ipbc,exps,E,mag,L_real)
                    end do

                    write(f_name, *) i, beta, mag, E/N

                end do
            end subroutine tarefaC1
```

No gráfico (**??**) 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 β para variações de $\Delta b = 0,001$.

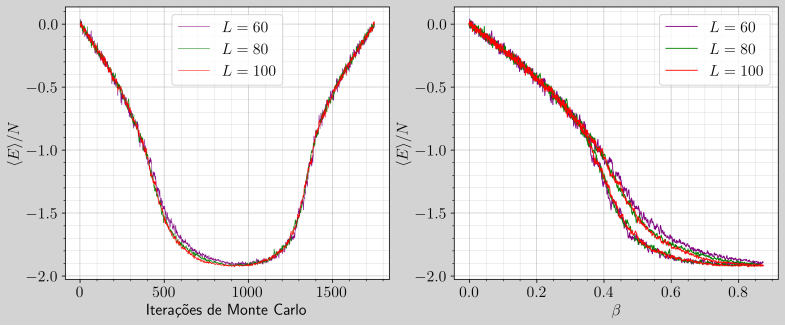

Figura 10: À esquerda energia média por spin por iterações de Monte Carlo e à direita em relação à β.

A figura (**??**) 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.

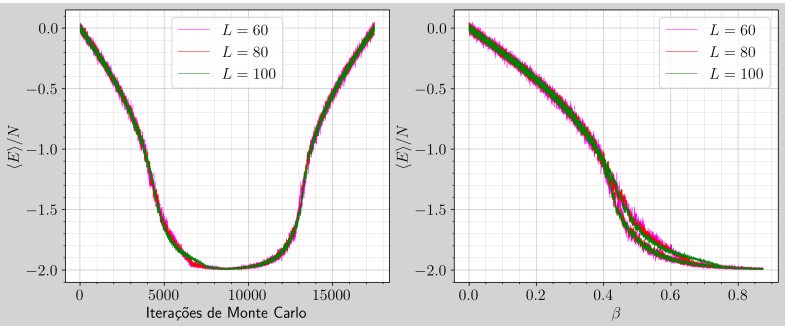

Figura 11: À esquerda energia média por spin por iterações de Monte Carlo e à direita em relação à β.

Podemos observar nos gráficos que a região de histerese

correspondem à um intervalo de β 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:

```
        dimension betas(1:5)
        parameter(betas = (/0.41, 0.44, 0.47, 0.51, 0.55/))

        open(1, file="saidas/tarefa-3/saida-tarefa-C2-L60-b1.dat")
        open(2, file="saidas/tarefa-3/saida-tarefa-C2-L60-b2.dat")
        open(3, file="saidas/tarefa-3/saida-tarefa-C2-L60-b3.dat")
        open(4, file="saidas/tarefa-3/saida-tarefa-C2-L60-b4.dat")
        open(5, file="saidas/tarefa-3/saida-tarefa-C2-L60-b5.dat")

        do i = 1, 5
            call tarefaC2(60, betas(i), i)
            close(1)
        end do

        open(1, file="saidas/tarefa-3/saida-tarefa-C2-L80-b1.dat")
        open(2, file="saidas/tarefa-3/saida-tarefa-C2-L80-b2.dat")
        open(3, file="saidas/tarefa-3/saida-tarefa-C2-L80-b3.dat")
        open(4, file="saidas/tarefa-3/saida-tarefa-C2-L80-b4.dat")
        open(5, file="saidas/tarefa-3/saida-tarefa-C2-L80-b5.dat")

        do i = 1, 5
            call tarefaC2(80, betas(i), i)
            close(1)
        end do

        open(1, file="saidas/tarefa-3/saida-tarefa-C2-L100-b1.dat")
        open(2, file="saidas/tarefa-3/saida-tarefa-C2-L100-b2.dat")
        open(3, file="saidas/tarefa-3/saida-tarefa-C2-L100-b3.dat")
        open(4, file="saidas/tarefa-3/saida-tarefa-C2-L100-b4.dat")
        open(5, file="saidas/tarefa-3/saida-tarefa-C2-L100-b5.dat")

        do i = 1, 5
            call tarefaC2(100, betas(i), i)
            close(1)
        end do
        end
        subroutine tarefaC2(L_real, beta, fname)
!               Tarefa B - Recozimento e quenching
            implicit integer(f-f)
            implicit real(j-j, m-m)
            parameter(L = 100)
            dimension exps(-4:4)
            byte lattice(1:L, 1:L)
            ! periodic boundary conditions
            dimension ipbc(0:L+1)

            do i = 1, L_real
                ipbc(i) = i
            end do

            ipbc(0) = L_real
```

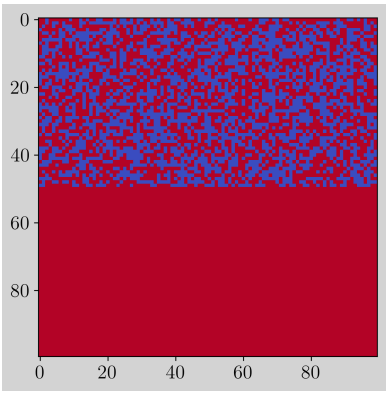

Figura 12: Configuração inicial para dinâmica utilizada na medida de temperatura crítica.

```
            ipbc(L_real+1) = 1

            N = L_real * L_real

            mag = 0.0d0

            call srand(L_real * 392)

            ! half ordered / half random.
            call initialize_lattice(lattice, L_real, L_real)
            call initialize_random_lattice(lattice,  L_real/2, L_real)

            open(99, file = "saidas/tarefa-3/saida-tarefa-C2-conf.dat")
            call write_lattice(lattice, L_real, 99)
            close(99)

            call total_magnetization(lattice, mag, L_real)

            ! initial energy
            E = H_0(lattice, ipbc, L_real)
            dbeta = 0.01
            write(fname, *) 0, E/N
            do i = 1, 3000
                call define_exponentials(exps, beta)
                do k = 1 , N
                    call flip_spin(lattice,ipbc,exps,E,mag,L_real)
                end do
                write(fname, *) i, E/N
            end do
        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 β discutido antes, isto é, $0,4 < \beta < 0.6$. A imagem (**??**) mostra a configuração inicial do sistema. Foram escolhidos alguns valores de β para executar a dinâmica de Monte Carlo.

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

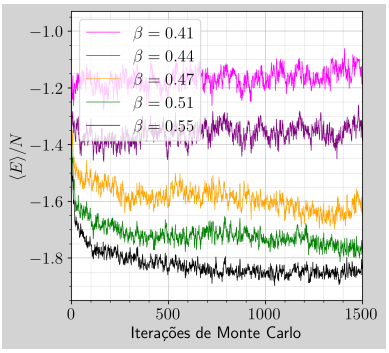

Figura 13: Dinâmica para L=80.

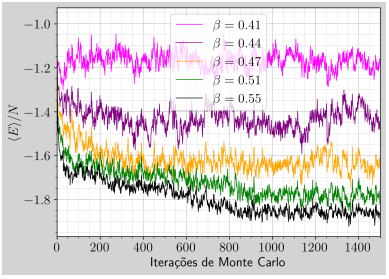

Figura 14: Dinâmica para L=60.

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 β relacionado à temperatura crítica $T_c = 1/\beta_c \approx 2,27$.

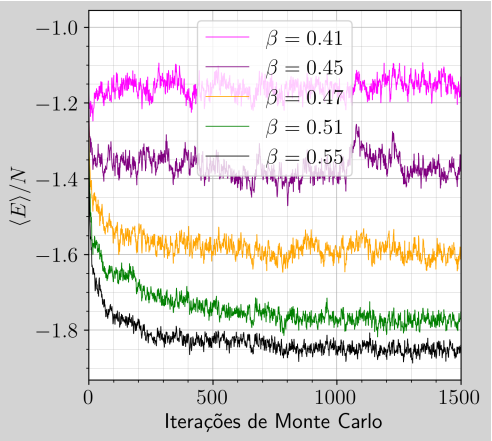

Figura 15: Dinâmica para L=100.

Além disso, pela (**??**) 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:

```
    implicit integer(f-f)
    implicit real(m-m)
    parameter(L = 100)
    dimension exps(-4:4)
    byte lattice(1:L, 1:L)
    ! periodic boundary conditions
    dimension ipbc(0:L+1)
    ! this or using mod
    open(unit=1, file="saidas/tarefa-4/saida-tarefa-D.dat")
    open(unit=2, file="saidas/tarefa-4/saida-tarefa-MAG_T.dat")
    beta = 0.5
    call define_exponentials(exps, beta)
    do L_real = 4, 10
        print *, "L = ", L_real
        call srand(3519) ! /L_real+1)
        N = L_real * L_real
        ! setting ipbc
        do i = 1, L_real
```

```
            ipbc(i) = i
        end do
        ipbc(0) = L_real
        ipbc(L_real+1) = 1
        mag = 0.0e0
        call initialize_random_lattice(lattice, L_real, L_real)
        call total_magnetization(lattice, mag, L_real)
        n_inversions = 10000
        n_curr = 0
        n_time = 0
        do while(n_curr < n_inversions)
            mag_prev = mag
            do i = 1, N
                call flip_spin(lattice, ipbc, exps, E, mag, L_real)
            end do
            n_time = n_time + 1
            ! Fazer o gráfico da magnetização aqui.
            if(mag_prev * mag < 0) then
                t_mean = t_mean + n_time
                n_time = 0
                n_curr = n_curr + 1
            end if
        end do
        write(2, *) t_mean, mag
        t_mean = t_mean / n_inversions
        write(1, *) L_real, t_mean
    end do
    close(1)
    end
```

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

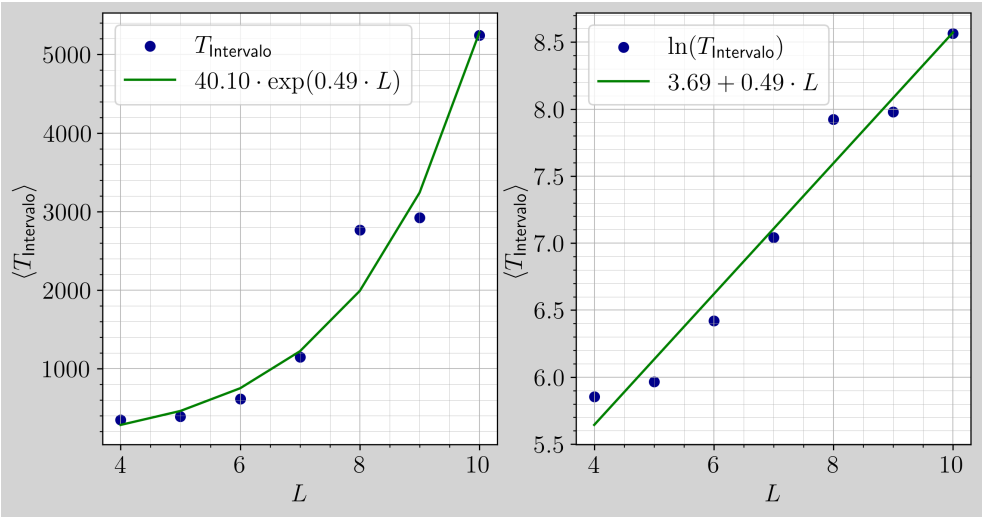

Figura 16: Gráfico do crescimento do intervalo $\langle T_{\text{intervalo}} \rangle$ em função de L e ajuste linear.

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

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 equilibrio mas a magnetização tem comportamento não usual. Aumentando

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