Outils, modélisation et simulation en calcul numérique – Corrigé série 10

24 mai 2005

Exercice 1.
L'équation maîtresse de la donnée peut s'écrire sous la forme

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}p(\sigma_1,\ldots,\sigma_N,t) &= \sum_{j=1}^{N}\omega(-\sigma_j)p(\sigma_1,\ldots,\sigma_{j-1},-\sigma_j,\sigma_{j+1},\ldots,\sigma_N,t)\\ &\quad -\sum_{j=1}^{N}\omega(\sigma_j)p(\sigma_1,\ldots,\sigma_{j-1},\sigma_j,\sigma_{j+1},\ldots,\sigma_N,t),\\ &= -\sum_{j=1}^{N}\sigma_j\sum_{\sigma'_j=\pm\sigma_j}\sigma'_j\omega_j(\sigma'_j)p(\sigma_1,\ldots,\sigma_{j-1},\sigma'_j,\sigma_{j+1},\ldots,\sigma_N,t). \qquad (1)\end{aligned}$$

En multipliant cette expression par σ et en sommant sur toutes les configurations $\{\sigma\}$ il vient

$$\frac{\mathrm{d}}{\mathrm{d}t}\sum_{\{\sigma\}}\sigma_k p(\sigma_1,\ldots,\sigma_N,t) = -\sum_{\{\sigma\}}\sum_{j=1}^{N}\sigma_k\sigma_j\sum_{\sigma'_j=\pm\sigma_j}\sigma'_j\omega_j(\sigma'_j)p(\sigma_1,\ldots,\sigma_{j-1},\sigma'_j,\sigma_{j+1},\ldots,\sigma_N,t). \qquad (2)$$

En séparant les termes $j=k$ des autres, il vient

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}t}\sum_{\{\sigma\}}\sigma_k p(\sigma_1,\ldots,\sigma_N,t) &= -\sum_{\{\sigma\}}\sigma_k\sigma_k\sum_{\sigma'_k=\pm\sigma_k}\sigma'_k\omega(\sigma'_k)p(\sigma_1,\ldots,\sigma_{k-1},\sigma'_k,\sigma_{k+1},\ldots,\sigma_N,t)\\ &\quad -\sum_{\{\sigma\}}\sum_{j\neq k}^{N}\sigma_k\sigma_j\sum_{\sigma'_j=\pm\sigma_j}\sigma'_j\omega(\sigma'_j)p(\sigma_1,\ldots,\sigma_{j-1},\sigma'_j,\sigma_{j+1},\ldots,\sigma_N,t). \qquad (3)\end{aligned}$$

Le second terme du membre de droite de l'Eq. (3) est impair en σ_j et donc sa contribution est nulle. Les deux contributions de la somme sur $\sigma'_k=\pm\sigma_k$ dans le premier terme du membre de droite de l'Eq. (3) engendrent un facteur 2 par une nouvelle définition des indices de sommation $\sigma_k\to-\sigma_k$. Ainsi

$$\frac{\mathrm{d}}{\mathrm{d}t}\sum_{\{\sigma\}}\sigma_k p(\sigma_1,\ldots,\sigma_N,t) = -2\sum_{\{\sigma\}}\sigma_k\omega_k(\sigma_k)p(\sigma_1,\ldots,\sigma_N,t), \qquad (4)$$

et en utilisant la définition de $q_k(t)=\langle\sigma_k\rangle$ il vient

$$\boxed{\frac{\mathrm{d}}{\mathrm{d}t}q_k(t) = -2\left\langle\sigma_k(t)\omega_k[\sigma_k(t)]\right\rangle.} \qquad (5)$$

De façon similaire, en multipliant l'équation maîtresse (1) par $\sigma_j\sigma_k$, $j\neq k$, en sommant sur toutes les configurations, et en en prenant la trace, on obtient le résultat cherché :

$$\boxed{\sigma_{ik}(t) = \langle\sigma_i(t)\sigma_k(t)\rangle = \sum_{\{\sigma\}}\sigma_i\sigma_k p(\sigma_1,\ldots,\sigma_N,t).} \qquad (6)$$

Soulignons que les équations du mouvement ci-dessus sont valables pour toute probabilité de transition et pour toute dimension.

Exercice 2.
En une dimension, l'état μ sera noté s_i et l'autre état $-s_i$. Le taux de transition s'écrit

$$\omega(s_i \to -s_i) = \frac{\exp[-\beta E(s_i)]}{\exp[-\beta E(s_i)] + \exp[-\beta E(-s_i)]}, \tag{7}$$

où l'énergie d'une configuration en champ nul dans la phase ferromagnétique ($J > 0$) avec couplage aux plus proches voisins en dimension 1 s'écrit

$$\begin{aligned}
-\beta E(s_j) &= -\beta \left[-J \frac{1}{2} \sum_{\langle k,l \rangle} s_k s_l \right] \\
&= \beta J \sum_{k=1}^{N-1} s_k s_{k+1} \\
&= \beta J \left[\left(\sum_{\substack{k=1 \\ k \neq \{i-1,i\}}}^{N-1} s_k s_{k+1} \right) + s_{i-1} s_i + s_i s_{i+1} \right] \\
&= \beta J \overline{E} + \beta J s_i (s_{i-1} + s_{1+1}),
\end{aligned} \tag{8}$$

où on a utilisé des conditions aux bords périodiques et on a noté $\langle k, l \rangle$ pour indiquer une somme sur les plus proches voisins. $\overline{E}$ est donc l'énergie du système sans le spin s_i. Ainsi

$$\exp(-\beta E) = \exp\left(\beta J \overline{E}\right) \exp\left[\beta J s_i (s_{i+1} + s_{i-1})\right], \tag{9}$$

que l'on remplace dans l'Eq. (7) en faisant usage de la relation $s_i = \pm 1 \ \forall i$, pour obtenir

$$\begin{aligned}
\omega(s_i \to -s_i) &= \frac{\exp\left(\beta J \overline{E}\right) \exp\left[-\beta J s_i (s_{i+1} + s_{i-1})\right]}{\exp\left(\beta J \overline{E}\right) \exp\left[-\beta J s_i (s_{i+1} + s_{i-1})\right] + \exp\left(\beta J \overline{E}\right) \exp\left[\beta J s_i (s_{i+1} + s_{i-1})\right]} \\
&= \frac{\exp\left[-\beta J s_i (s_{i+1} + s_{i-1})\right]}{2 \cosh\left[-\beta J (s_{i+1} + s_{i-1})\right]} \\
&= \frac{\cosh\left[\beta J (s_{i-1} + s_{i+1})\right]}{2 \cosh\left[\beta J (s_{i-1} + s_{i+1}\right]} \left[1 - \frac{1}{2} s_i \left(s_{i-1} + s_{i+1}\right) \tanh(2\beta J) \right] \\
&= \frac{1}{2} \left[1 - \frac{1}{2} s_i (s_{i-1} + s_{i+1}) \gamma \right], \qquad \gamma = \tanh(2\beta J),
\end{aligned} \tag{10}$$

ce qui est le résultat cherché.

Exercice 3.

i) Programme Matlab :

```
clear; %just to view in realtime the spins
L=100; %system size L^2
steps=1000; %Monte Carlo steps
beta=0.40; %interac: J\beta. beta_c=0.44069: infinite system
dynamics=1; %=1: Metropolis; else: Glauber
nshow=5; %displays every nsave time steps
colormap(bone);
spin=ones(L,L);
m=[L 1:L-1];
p=[2:L 1];
k=0;
for i=1:steps
  for j=1:L^2
    x=floor(L*rand(1,1))+1;
    y=floor(L*rand(1,1))+1;
    dE=2*spin(x,y)*(spin(p(x),y) + spin(m(x),y) + spin(x,p(y)) + spin(x,m(y)));
    if (dynamics ~= 1) %Glauber
      prob = exp(-beta*dE);
      prob = prob/(1+prob);
    else %Metropolis
      if (exp(-beta*dE) < 1),prob=exp(-beta*dE);else,prob = 1;end
    end
    if (rand(1,1) < prob),spin(x,y)=-spin(x,y);end
  end
  if ((mod(i,nshow)==0) | (i==1))
    imagesc(spin);
    axis square off;
    drawnow;
  end
end
```

Programme Fortran 90 :

```
module fonctions
implicit none
integer, parameter :: L=100 !system size = L^2
integer, parameter :: steps=10000 !number of Monte-Carlo steps
real, parameter :: beta=0.42 !interac: J\beta. beta_c=0.44069: infinite system
integer, parameter :: dynamics=1 !=1: Metropolis; else: Glauber
integer, parameter :: nsave=50 !saving data every nsave time steps
character*(*), parameter :: filename1 = 's10-3i-m.txt'
character*(*), parameter :: filename2 = 's10-3i-spins.txt'
contains
!---------------------------------------------------------------------
```

```
FUNCTION ran2(idum) !Initialize idum with negative integer.
INTEGER idum,IM1,IM2,IMM1,IA1,IA2,IQ1,IQ2,IR1,IR2,NTAB,NDIV
REAL :: ran2,AM,EPS,RNMX
PARAMETER (IM1=2147483563,IM2=2147483399,AM=1./IM1,IMM1=IM1-1,IA1=40014,&
          &IA2=40692,IQ1=53668,IQ2=52774,IR1=12211,IR2=3791,NTAB=32,&
          &NDIV=1+IMM1/NTAB,EPS=1.2e-7,RNMX=1.-EPS)
INTEGER :: idum2,j,k,iv(NTAB),iy
SAVE iv,iy,idum2
DATA idum2/123456789/, iv/NTAB*0/, iy/0/
if(idum.le.0)then
  idum=max(-idum,1)
  idum2=idum
  do j=NTAB+8,1,-1
    k=idum/IQ1
    idum=IA1*(idum-k*IQ1)-k*IR1
    if(idum.lt.0)idum=idum+IM1
    if(j.le.NTAB)iv(j)=idum
  enddo
  iy=iv(1)
endif
k=idum/IQ1
idum=IA1*(idum-k*IQ1)-k*IR1
if(idum.lt.0)idum=idum+IM1
k=idum2/IQ2
idum2=IA2*(idum2-k*IQ2)-k*IR2
if(idum2.lt.0)idum2=idum2+IM2
j=1+iy/NDIV
iy=iv(j)-idum2
iv(j)=idum
if(iy.lt.1)iy=iy+IMM1
ran2=min(AM*iy,RNMX)
return
end function ran2
!-----------------------------------------------------------------------
end module fonctions
program s10ex3i
use fonctions
implicit none
integer :: idum,i,j,x,y,xp,xm,yp,ym,magnet
real :: prob,dE
integer, dimension(0:L-1,0:L-1) :: spin
idum = -4856
spin=1
magnet=sum(spin)
open(unit=1,file=filename1)
close(1,status='DELETE')
open(unit=1,file=filename1,position='APPEND')
write(1,*) 0,real(magnet)/real(L**2)
```

```
close(1)
do i=1,steps
  do j=1,L**2
    x = int(L*ran2(idum))
    y = int(L*ran2(idum))
    xp = mod(x+1,L)
    xm = mod(x-1+L,L)
    yp = mod(y+1,L)
    ym = mod(y-1+L,L)
    dE = real(2*spin(x,y)*(spin(xp,y) + spin(xm,y) + spin(x,yp) + spin(x,ym)))
    if (dynamics .neq. 1) then !Glauber
      prob = exp(-beta*dE)
      prob = prob/(1.+prob)
    else !Metropolis
      if (exp(-beta*dE) < 1.) then
        prob = exp(-beta*dE)
      else
        prob = 1.
      end if
    end if
    if (ran2(idum) < prob) then
      spin(x,y) = -spin(x,y)
      magnet = magnet + 2*spin(x,y)
    end if
  end do
  if ((mod(i,nsave)==0) .or. (i==1)) then
    open(unit=1,file=filename1,position='APPEND')
    write(1,*) i,real(magnet)/real(L**2)
    close(1)
    open(unit=1,file=filename2)
    do x=0,L-1
      do y=0,L-1
        if (spin(x,y) == 1) write(1,*) x+1,y+1
      end do
    end do
    close(1)
  end if
end do
end program s10ex3i
```

La Fig. 1 représente l'aimantation moyenne $\langle m \rangle$ en fonction du nombre de pas de temps τ.
On constate qu'après un temps de relaxation de l'ordre de 500 itérations, l'aimantation devient nulle en moyenne temporelle. La Fig. 2 représente la configuration de spins pour deux valeurs différentes de β.
On constate que pour $\beta = 0.4$ l'aimantation est en moyenne nulle, tandis que pour $\beta = 0.48$ elle est non nulle. En effet, pour un système de taille infinie la température critique est $\beta_c = 0.44069\ldots$ telle que pour tout $\beta > \beta_c$ il existe une aimantation spontanée non nulle.

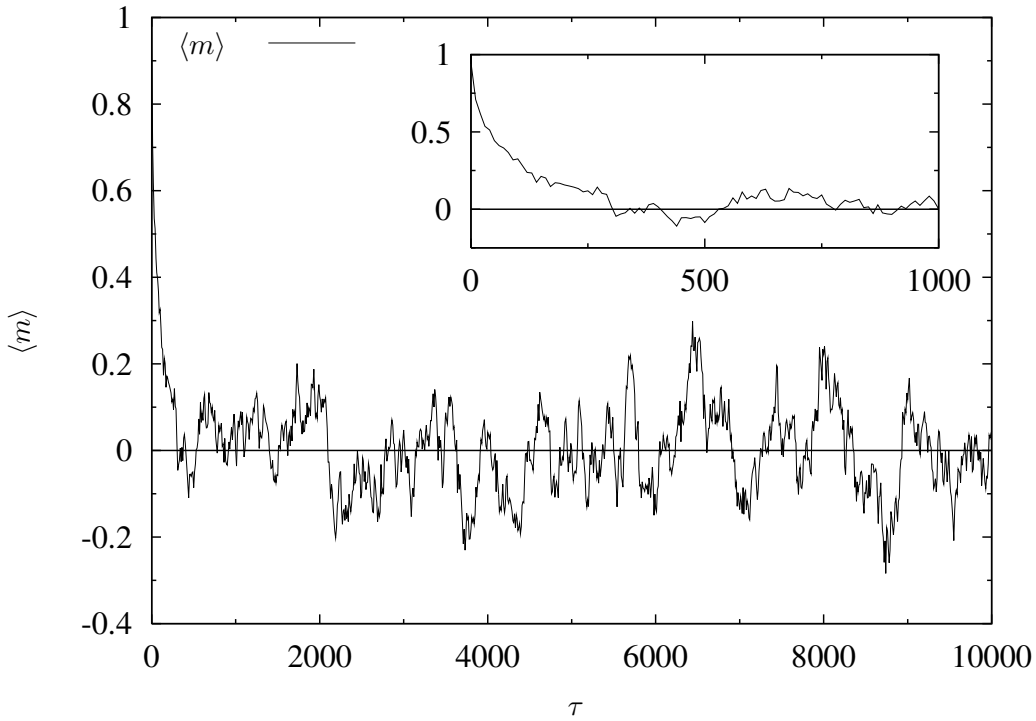

FIG. 1 – Aimantation moyenne $\langle m \rangle$ en fonction du nombre de pas de temps τ pour une dynamique de Metropolis. Un pas de temps est défini par une actualisation de chaque spin en moyenne, c'est-à-dire L^2 itérations. La taille du système est $L^2 = 10^4$, la température est $\beta = 0.4$. La condition initiale est telle que $\langle m \rangle = 1$.

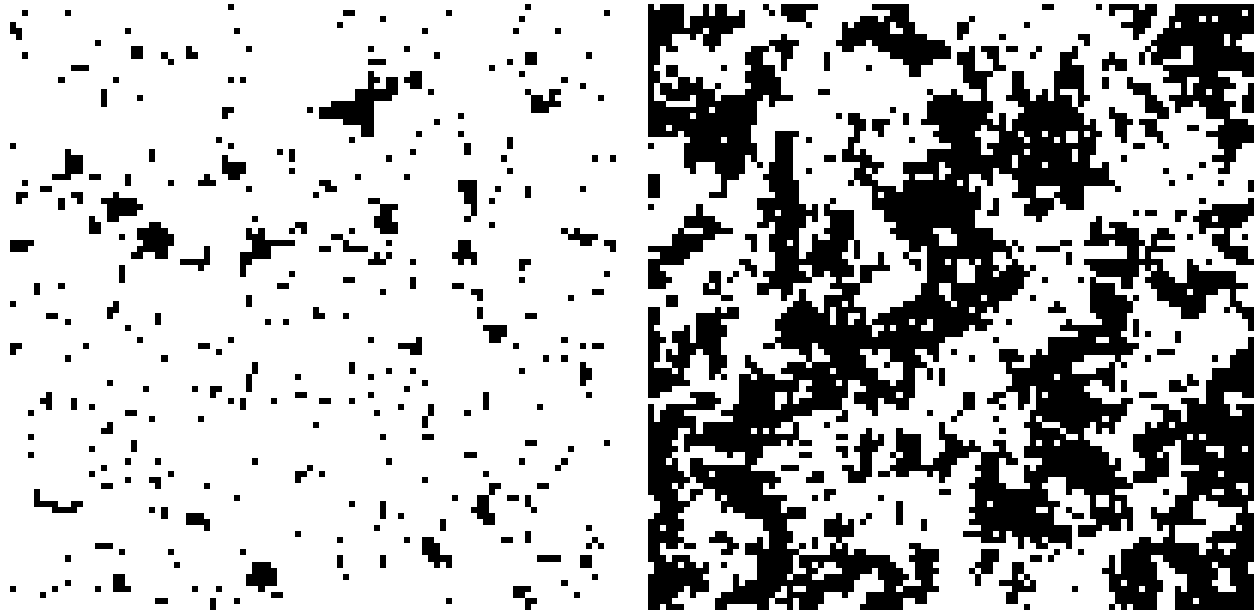

FIG. 2 – Configuration des spins après $\tau = 10^3$ pas de temps pour un système de taille $L^2 = 10^4$ et une condition initiale telle que $\langle m \rangle = 1$. L'image de gauche est obtenue avec $\beta = 0.48$, et l'image de droite avec $\beta = 0.40$.

ii) Programm Fortran 90 :

```
module fonctions
implicit none
integer, parameter :: L=50 !system size = L^2
integer, parameter :: steps=50000 !number of Monte-Carlo steps
real, parameter :: beta_min=0.30 !beta_c=0.44069: infinite system
real, parameter :: beta_max=0.60
integer, parameter :: npoints=300 !#points for \beta \in [\beta_min,\beta_max]
integer, parameter :: dynamics=1 !=1: Metropolis; else: Glauber
character*(*), parameter :: filename = 's10-3ii.txt'
contains
!-----------------------------------------------------------------------
FUNCTION ran2(idum) !Initialize idum with negative integer.
INTEGER idum,IM1,IM2,IMM1,IA1,IA2,IQ1,IQ2,IR1,IR2,NTAB,NDIV
REAL :: ran2,AM,EPS,RNMX
PARAMETER (IM1=2147483563,IM2=2147483399,AM=1./IM1,IMM1=IM1-1,IA1=40014,&
          &IA2=40692,IQ1=53668,IQ2=52774,IR1=12211,IR2=3791,NTAB=32,&
          &NDIV=1+IMM1/NTAB,EPS=1.2e-7,RNMX=1.-EPS)
INTEGER :: idum2,j,k,iv(NTAB),iy
SAVE iv,iy,idum2
DATA idum2/123456789/, iv/NTAB*0/, iy/0/
if(idum.le.0)then
  idum=max(-idum,1)
  idum2=idum
  do j=NTAB+8,1,-1
    k=idum/IQ1
    idum=IA1*(idum-k*IQ1)-k*IR1
    if(idum.lt.0)idum=idum+IM1
    if(j.le.NTAB)iv(j)=idum
  enddo
  iy=iv(1)
endif
k=idum/IQ1
idum=IA1*(idum-k*IQ1)-k*IR1
if(idum.lt.0)idum=idum+IM1
k=idum2/IQ2
idum2=IA2*(idum2-k*IQ2)-k*IR2
if(idum2.lt.0)idum2=idum2+IM2
j=1+iy/NDIV
iy=iv(j)-idum2
iv(j)=idum
if(iy.lt.1)iy=iy+IMM1
ran2=min(AM*iy,RNMX)
return
end function ran2
!-----------------------------------------------------------------------
end module fonctions
```

```
program s10ex3ii
use fonctions
implicit none
integer :: idum,i,j,k,x,y,xp,xm,yp,ym,magnet
real :: prob,dE,inc,beta
integer, dimension(0:L-1,0:L-1) :: spin
real, dimension(1:steps) :: mt
idum = -4856
open(unit=1,file=filename)
close(1,status='DELETE')
inc=(beta_max-beta_min)/real(npoints)
do k=0,npoints
  beta=beta_min+k*inc
  spin=1
  magnet=sum(spin)
  do i=0,steps
    do j=1,L**2
      x = int(L*ran2(idum))
      y = int(L*ran2(idum))
      xp = mod(x+1,L)
      xm = mod(x-1+L,L)
      yp = mod(y+1,L)
      ym = mod(y-1+L,L)
      dE = real(2*spin(x,y)*(spin(xp,y) + spin(xm,y) + spin(x,yp) + spin(x,ym)))
      if (dynamics .neq. 1) then !Glauber
        prob = exp(-beta*dE)
        prob = prob/(1.+prob)
      else !Metropolis
        if (exp(-beta*dE) < 1.) then
          prob = exp(-beta*dE)
        else
          prob = 1.
        end if
      end if
      if (ran2(idum) < prob) then
        spin(x,y) = -spin(x,y)
        magnet = magnet + 2*spin(x,y)
      end if
    end do
    mt(i)=real(magnet)/real(L**2)
  end do
  open(unit=1,file=filename,position='APPEND')
  write(1,*) beta,sum(mt(1000:steps))/real(steps-1000)
  close(1)
end do
end program s10ex3ii
```

La Fig. 3 représente une étude de l'aimantation moyenne $\langle m \rangle$ en fonction de la température β.

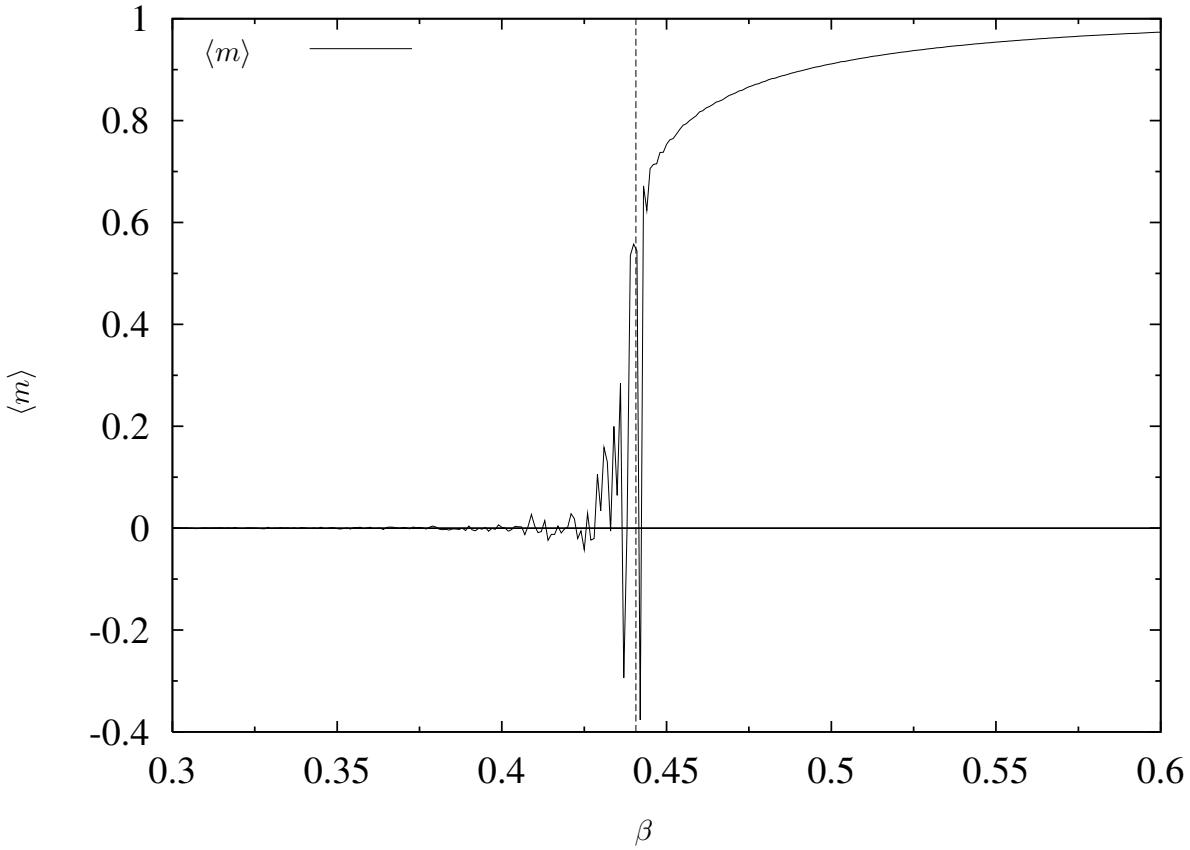

FIG. 3 – Aimantation moyenne $\langle m \rangle$ en fonction de la température $\beta = (k_B T)^{-1}$ pour un système de taille $L^2 = 10^4$ et une dynamique de Metropolis. Chaque point est obtenu par moyenne sur 5×10^4 valeurs de l'aimantation. Pour un système infini, la valeur critique est $\beta_c = 0.44069\ldots$, représentée par la ligne vertical discontinue.

A grande température [$\beta < \beta_c(L)$, où $\beta_c(L < \infty) \neq \beta_c(L = \infty) = 0.44069\ldots$ à cause des effets de taille finie] le système est désordonné et la valeur moyenne de l'aimantation est nulle. En se rapprochant du point critique, l'amplitude des fluctuations augmente. En $\beta = \beta_c(L)$ se produit une transition de phase, et pour $\beta > \beta_c(L)$ l'aimantation moyenne devient non nulle. Notons que la condition initiale est choisie telle que $\langle m \rangle = 1$, et donc pour $\beta > \beta_c(L)$ l'état d'aimantation non nulle sera tel que $\langle m \rangle > 0$. En choisissant une condition initiale telle que $\langle m \rangle = 0$, on obtiendrait de façon équiprobable une aimantation non nulle positive ou négative.
La Fig. 3 représente l'aimantation moyenne $\langle m \rangle$ en fonction de la température β pour deux systèmes de taille plus petite, $L^2 = 2'500$ et $L^2 = 400$.

On constate d'une part que l'amplitude des fluctuations est plus grande (comparer les échelles verticales), et d'autre part que la valeur du point critique β_c devient supérieure à celle du point critique du système infini.

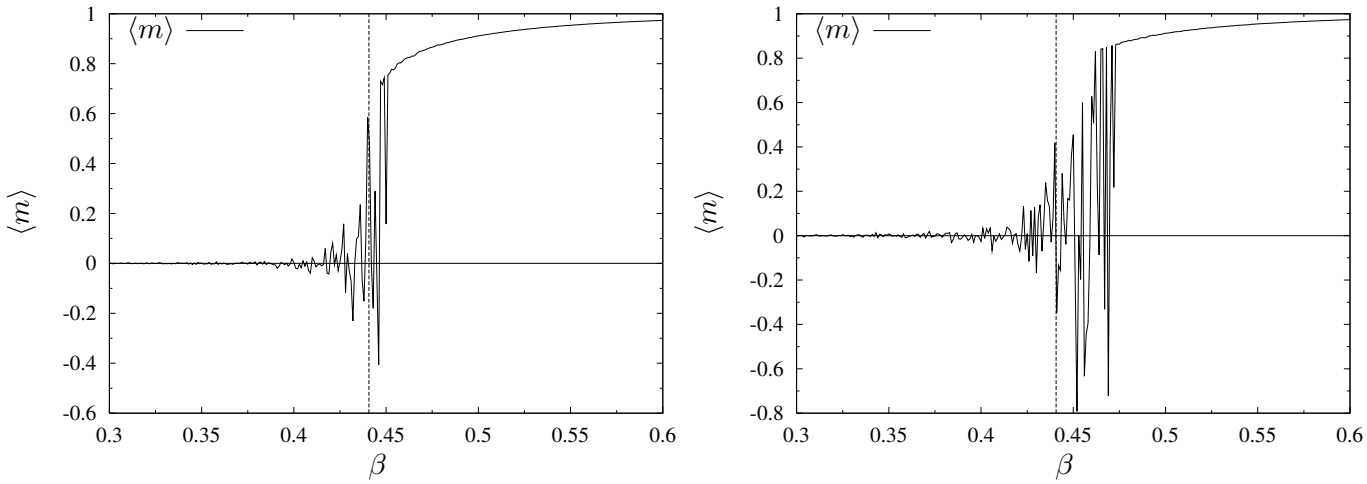

FIG. 4 – Aimantation moyenne $\langle m \rangle$ en fonction de la température $\beta = (k_B T)^{-1}$ pour un système de taille $L^2 = 2'500$ (image de gauche), $L^2 = 400$ (image de droite), et une dynamique de Metropolis. Chaque point est obtenu par moyenne sur 5×10^4 valeurs de l'aimantation. Pour un système infini, la valeur critique est $\beta_c = 0.44069\ldots$, représentée par la ligne vertical discontinue.

iii) Programme Fortran 90 :

```
module fonctions
implicit none
integer, parameter :: L=100 !system size = L^2
integer, parameter :: steps=10**9 !number of Monte-Carlo steps
real, parameter :: beta=0.46 !interac: J\beta. beta_c=0.44069: infinite system
integer, parameter :: dynamics=1 !=1: Metropolis; else: Glauber
integer, parameter :: points=100 !resolution of the histogram in [0,1]
character*(*), parameter :: filename = 's10-3iii.txt'
contains
!------------------------------------------------------------------------
FUNCTION ran2(idum) !Initialize idum with negative integer.
INTEGER idum,IM1,IM2,IMM1,IA1,IA2,IQ1,IQ2,IR1,IR2,NTAB,NDIV
REAL :: ran2,AM,EPS,RNMX
PARAMETER (IM1=2147483563,IM2=2147483399,AM=1./IM1,IMM1=IM1-1,IA1=40014,&
          &IA2=40692,IQ1=53668,IQ2=52774,IR1=12211,IR2=3791,NTAB=32,&
          &NDIV=1+IMM1/NTAB,EPS=1.2e-7,RNMX=1.-EPS)
INTEGER :: idum2,j,k,iv(NTAB),iy
SAVE iv,iy,idum2
DATA idum2/123456789/, iv/NTAB*0/, iy/0/
if(idum.le.0)then
  idum=max(-idum,1)
  idum2=idum
  do j=NTAB+8,1,-1
    k=idum/IQ1
    idum=IA1*(idum-k*IQ1)-k*IR1
    if(idum.lt.0)idum=idum+IM1
    if(j.le.NTAB)iv(j)=idum
```

```
  enddo
  iy=iv(1)
endif
k=idum/IQ1
idum=IA1*(idum-k*IQ1)-k*IR1
if(idum.lt.0)idum=idum+IM1
k=idum2/IQ2
idum2=IA2*(idum2-k*IQ2)-k*IR2
if(idum2.lt.0)idum2=idum2+IM2
j=1+iy/NDIV
iy=iv(j)-idum2
iv(j)=idum
if(iy.lt.1)iy=iy+IMM1
ran2=min(AM*iy,RNMX)
return
end function ran2
!-----------------------------------------------------------------------
end module fonctions
program s10ex3iii
use fonctions
implicit none
integer :: idum,i,j,x,y,xp,xm,yp,ym,magnet
real :: prob,dE
integer, dimension(0:L-1,0:L-1) :: spin
integer, dimension(1:points) :: histo
idum = -4856
histo=0.
spin=1
magnet=sum(spin)
do i=1,steps !a few point in a given run
  do j=1,L**2
    x = int(L*ran2(idum))
    y = int(L*ran2(idum))
    xp = mod(x+1,L)
    xm = mod(x-1+L,L)
    yp = mod(y+1,L)
    ym = mod(y-1+L,L)
    dE = real(2*spin(x,y)*(spin(xp,y) + spin(xm,y) + spin(x,yp) + spin(x,ym)))
    if (dynamics .neq. 1) then !Glauber
      prob = exp(-beta*dE)
      prob = prob/(1.+prob)
    else !Metropolis
      if (exp(-beta*dE) < 1.) then
        prob = exp(-beta*dE)
      else
        prob = 1.
      end if
    end if
```

```
    if (ran2(idum) < prob) then
      spin(x,y) = -spin(x,y)
      magnet = magnet + 2*spin(x,y)
    end if
  end do
  if ((mod(i,L)==0) .and. (i>L**2)) then !decorr.time=L, relax.time=L^2
    j=int(real(points)*(abs(real(magnet))/real(L**2))+1.)
    if (magnet > 0) histo(j)=histo(j)+1
    prob=(1./real(points))*real(sum(histo(1:points-1))+sum(histo(2:points)))/2.
    open(unit=1,file=filename) !saving data
    do j=0,points-1
      write(1,*) (real(j)+0.5)*(1./real(points)),real(histo(j+1))/(2.*prob)
    end do
    close(1)
 end if
end do
end program s10ex3iii
```

La Fig. 5 représente la fonction de distribution de l'aimantation pour différentes valeurs de la température β. On constate que pour de grandes températures, soit $\beta < \beta_c$, le système est

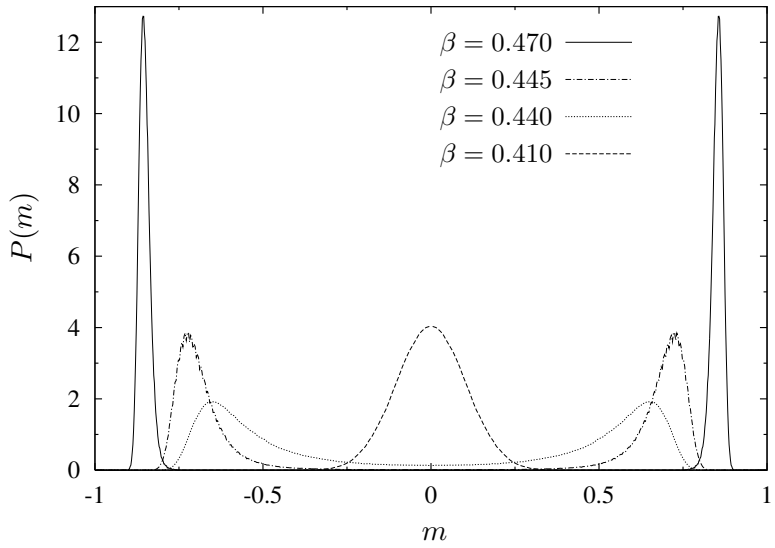

FIG. 5 – Fonction de distribution de l'aimantation $P(m)$ pour différentes températures β, pour un système de taille $L^2 = 10^4$, et pour une dynamique de Metropolis. La température du point critique du système infini est $\beta_c = 0.44069$.

désordonné et l'aimantation moyenne nulle. La fonction de distribution possède donc un unique pic centré à l'origine. Au fur et à mesure que la température décroît (donc β augmente) pour se rapprocher de la température critique β_c, la distribution s'élargit et la hauteur du pic central diminue. Enfin, pour de faibles températures ($\beta > \beta_c$) le système s'ordonne en deux phases d'aimantation non nulle $\pm m$, d'où l'apparition de deux pics qui se déplacent vers les extrémités du domaine de définition de la distribution lorsque la température diminue (la position de ces pics est donnée par la Fig. 3).