Rafał Świętek

Nr indeksu - 236668

Model Isinga – Metoda Monte Carlo

```
program ModelIsinga-sim
 implicit none
 integer d, i, j, k, a
 integer L, mcs, N
 parameter(L=200, mcs=50000)
 integer S(L,L), ni(L), pi(L)
 real ran1, magnetisation
 real R, dE
 real w, v, T, m, X, U, C
 d=-1
 a=1
 T=3.3
 open(11,file='magn.txt')
 call neighbour(L,ni,pi)
 call randomset(L,S)
 do k=1, mcs
    do i=1, L
       do j=1, L
   dE=2*S(i,j)*(S(ni(i),j)+S(pi(i),j))
   dE=dE+S(i,j)*S(i,ni(j))+S(i,pi(j))
          v = exp(-dE/T)
          w = min(1.0,v)
          R = ran1(d)
          if(R<=w) S(i,j) = -S(i,j)
       enddo
    enddo
     m = magnetisation(L,S)
     if(k>=30000) write(11,*) k, m
     write (*,*) k
 enddo
 call showmatrix(L,S)
 write(*,*) 'Done!'
 read(*,*)
end

subroutine randomset(k,A)
 integer k, A(k,k)
 real ran1, p
 integer d,i,j
 d = -1
 i = 1
 do i=1, k
   do j=1, k
       p = ran1(d)
       if(p<0.5) then
           A(i,j) = -1
       else
           A(i,j) = 1
       endif
    enddo
 enddo
 end

subroutine neighbour(k,ni,pi)
 integer k, ni(k), pi(k), i
 do i=1, k
   ni(i) = i+1
   pi(i) = i-1
 enddo
 ni(k)=1
 pi(1)=k
 end

function magnetisation(k,A)
 integer k, i, j, A(k,k), n
 real m, magnetisation
 n = k**2
 M = 0
 do i=1, k
   do j=1, k
      M = M + A(i,j)
   enddo
 enddo
 magnetisation = m/(n+0.0)
 return
 end

subroutine showmatrix(n, B)
 integer n, B(n,n)
 open(88,file='matrixL=200,T=3.3.txt')
 do j=1, n
    write(88,*)(B(i,j), i=1, n)
 enddo
 close(88)
 end

      FUNCTION ran1(idum)
      INTEGER idum,IA,IM,IQ,IR,NTAB,NDIV
      REAL ran1,AM,EPS,RNMX
      PARAMETER (IA=16807,IM=2147483647,
                 AM=1./IM,IQ=127773,IR=2836,
                 *NTAB=32,NDIV=1+(IM-1)/NTAB,
                 EPS=1.2e-7,RNMX=1.-EPS)
      INTEGER j,k,iv(NTAB),iy
      SAVE iv,iy
      DATA iv /NTAB*0/, iy /0/
      if (idum.le.0.or.iy.eq.0) then
        idum=max(-idum,1)
        do 11 j=NTAB+8,1,-1
          k=idum/IQ
          idum=IA*(idum-k*IQ)-IR*k
          if (idum.lt.0) idum=idum+IM
          if (j.le.NTAB) iv(j)=idum
11      continue
        iy=iv(1)
      endif
      k=idum/IQ
      idum=IA*(idum-k*IQ)-IR*k
      if (idum.lt.0) idum=idum+IM

      j=1+iy/NDIV
      iy=iv(j)
      iv(j)=idum
      ran1=min(AM*iy,RNMX)
      return
      END
```

Model Isinga – Metoda Monte Carlo

```
program ModelIsinga
        implicit none
        integer d, i, j, k
        integer L, mcs, a
        parameter(L=100, mcs=230000)
        integer S(L,L), ni(L), pi(L)
        real Hamiltonian,ran1,magnetisation,
               average
        real susceptibility, kumulantBindera,
               thermalcapacity
        real R, dE
        real w, v, T, m, mag, X, U, C, H
        real y1, x1, Tc
        real magn(mcs/100), Energy(mcs/100)
*        parameter(kB = 1.38e-23)
        d=-1
        T=1.8
        open(11,file='av-magnL=100.txt')
        open(12,file='PodatnoscL=100.txt')
        open(13,file='kumBindL=100.txt')
        open(14,file='poj.cieplnaL=100.txt')
        open(15,file='HamiltonianL=100.txt')
        call neighbour(L,ni,pi)
        call cleanarray(mcs/100,magn,Energy)
        call randomset(L,S)
 7      continue
         a = 1
         do k=1, mcs
           do i=1, L
              do j=1, L
          dE=2*S(i,j)*(S(ni(i),j)+S(pi(i),j))
          dE=dE+S(i,j)*S(i,ni(j))+S(i,pi(j))
               v = exp(-dE/T)
               w = min(1.0,v)
               R = ran1(d)
               if(R<=w) S(i,j) = -S(i,j)
              enddo
           enddo
         if(k>=30000.and.mod(k,100)==0) then
           magn(a) = abs(magnetisation(L,S))
           Energy(a) = Hamiltonian(L,S,ni,pi)
                 a = a + 1
          endif
         enddo
         a = a - 1
          m = average(a,magn,1)
          X = susceptibility(L,a,S,T,magn)
          U = kumulantBindera(L,a,S,magn)
          C = thermalcapacity(L,a,S,T,Energy)
          H = average(a,Energy,1)/L**2
          write(11,*) T, m
          write(12,*) T, X
          if(T>=2.0.and.T<=2.5)write(13,*)T,U
          write(14,*) T, C
          write(15,*) T, H
          T = T + 0.01
        if(T<=3.0) goto 7

        write(*,*) 'DONE!'
        PAUSE
       end

       subroutine randomset(k,A)
        integer k, A(k,k)
        real ran1, p
        integer d,i,j
        d = -1
        i = 1
        do i=1, k
          do j=1, k
             p = ran1(d)
             if(p<0.5) then
                 A(i,j) = -1
             else
                 A(i,j) = 1
             endif
          enddo
        enddo
        end

        subroutine neighbour(k,ni,pi)
         integer k, ni(k), pi(k), i
         do i=1, k
           ni(i) = i+1
           pi(i) = i-1
         enddo
         ni(k)=1
         pi(1)=k
         end
         subroutine cleanarray(n,p,q)
         integer n
         real p(n), q(n)
         do i=1, n
            p(i) = 0
            q(i) = 0
          enddo
         end
       function average(k,X,n)
         integer k, i, n
         real X(k)
         real average
         average = 0
         do i=1, k
          average = average + X(i)**n
         enddo
         average = average/(k+0.0)
         return
         end

        function Hamiltonian(k,A,ni,pi)
         integer k, i, j
         integer A(k,k), ni(k), pi(k)
         real H, Hamiltonian
         H=0
         do i=1, k
           do j=1,k
         H = H - A(i,j)*(A(ni(i),j)+A(pi(i),j)
                      +A(i,ni(j))+A(i,pi(j)))
           enddo
         enddo
         Hamiltonian = H/2
         return
         end
```

Model Isinga – Metoda Monte Carlo

```
function magnetisation(k,A)
  integer k, i, j, A(k,k), n
  real m, magnetisation
  n = k**2
  m = 0
  do i=1, k
    do j=1, k
       m = m + A(i,j)
    enddo
  enddo
  magnetisation = m/(n+0.0)
  return
  end

 function susceptibility(k,m,A,T,tab)
  integer k, i, j, A(k,k), n, m
  real T, tab(m), susceptibility
  real average
  n = k**2
 susceptibility = n/T*(average(m,tab,2)
           - average(m,tab,1)**2)
  return
  end

 function thermalcapacity(k,m,A,T,Etab)
  integer k, A(k,k), m
  real T, Etab(m)
  real thermalcapacity, average
  thermalcapacity=(average(m,Etab,2)-
         average(m,Etab,1)**2)/(T**2)
  thermalcapacity =thermalcapacity/k**2
  return
  end

 function kumulantBindera(k,m,A,mag)
  integer k, A(k,k), n
  real mag(m), kumulantBindera, average
  n = k**2
  kumulantBindera =
= 1-average(m,mag,4)/(3*average(m,mag,2)**2)
  return
  end
```

```
program Skalowanie
      implicit none
      real y1, x1, T, Tc, m10, m20, m40,
              m100
      integer L, i
      real Tempcritical
      open(10,file='av-magnL=10.txt')
      open(40,file='av-magnL=40.txt')
      open(100,file='av-magnL=100.txt')
      open(11,file='skaL=10.txt')
      open(41,file='skaL=40.txt')
      open(101,file='skaL=100.txt')
      Tc = Tempcritical(50)
      write(*,*) 'Tc = ',Tc
      PAUSE
      DO i=1, 100
        read(10,*) T, m10
        write(*,*) T, m10
        L = 10
        y1 = log(m10) + 0.125*log(L+0.0)
        x1 = log(abs(1-T/Tc)*(L+0.0))
        write(11,*) x1, y1
      enddo
      close(11)
      close(10)
      DO i=1, 100
        read(40,*) T, m40
        write(*,*) T, m40
        L = 40
        y1 = log(m40) + 0.125*log(L+0.0)
        x1 = log(abs(1-T/Tc)*(L+0.0))
        write(41,*)x1, y1
      enddo
      close(40)
      close(41)

      DO i=1, 100
        read(100,*) T, m100
        write(*,*) T, m100
        L = 100
        y1 = log(m100) + 0.125*log(L+0.0)
        x1 = log(abs(1-T/Tc)*(L+0.0))
        write(101,*) x1, y1
 enddo
 close(100)
 close(101)
end

Function Tempcritical(k)
 integer i, k, error
 real U100(k), U10(k), U30(k)
 real Temp(k)
 real min1, min2, Tempcritical
 real a, b, T1, T2
 open(21,file='kumBindL=10.txt')
 open(22,file='kumBindL=30.txt')
 open(23,file='kumBindL=100.txt')
 i = 1
 DO i=1, k
   read(21,*) Temp(i), U10(i)
   read(22,*) x, U30(i)
   read(23,*) x, U100(i)
 enddo
 close(21)
 close(22)
 close(23)
 min1 = abs(1 - U100(20)/U30(20))
 min2 = abs(1 - U30(20)/U10(20))
 do i=21,k
    a = abs(1 - U100(i)/U30(i))
    b = abs(1 - U30(i)/U10(i))
    if(a < min1) then
       min1 = a
       T1 = Temp(i)
    endif
    if(b < min2) then
       min2 = b
       T2 = Temp(i)
    endif
 enddo
 write(*,*) T1, T2
 Tempcritical = (T1 + T2)/2.0
 return
 end
```

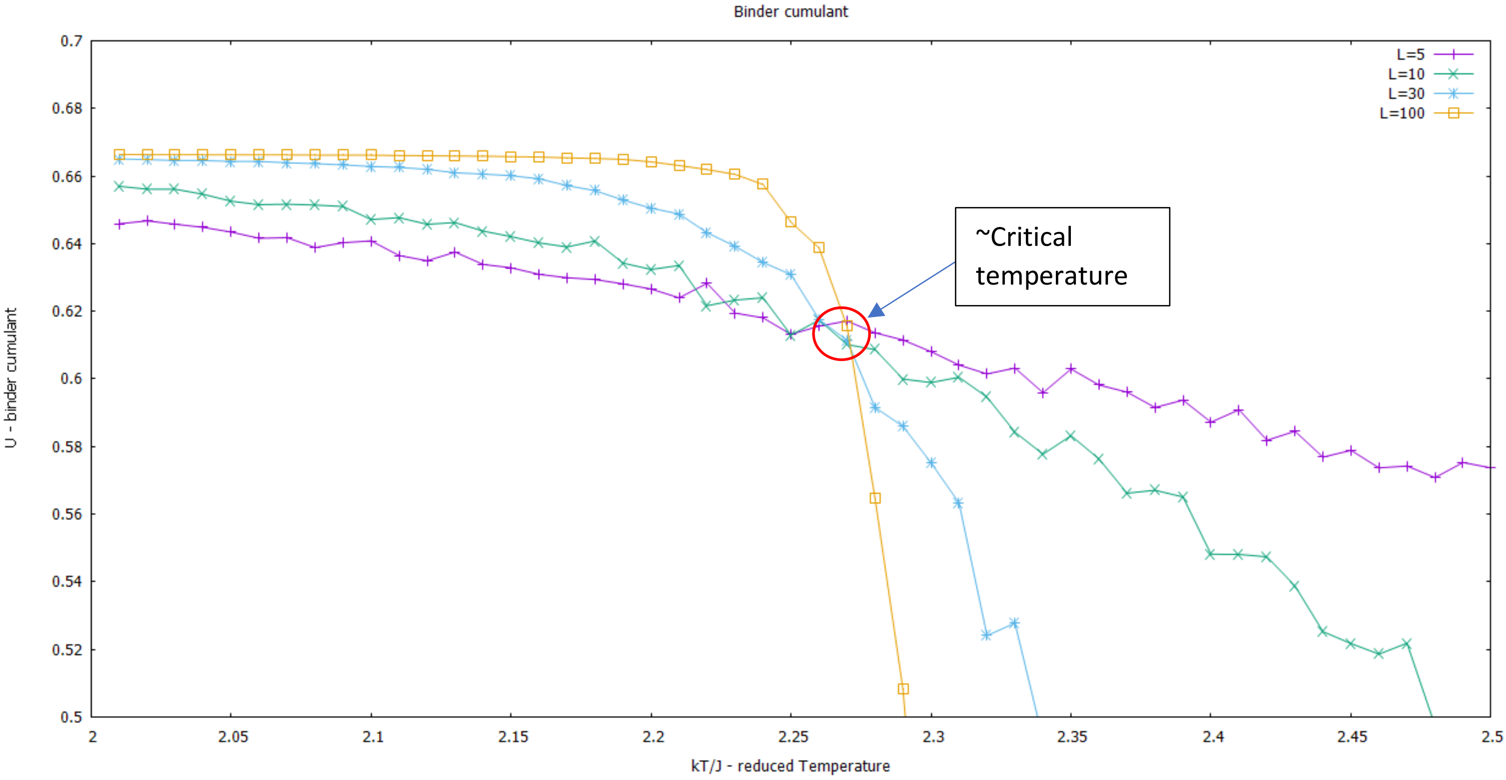
Model Isinga – Metoda Monte Carlo

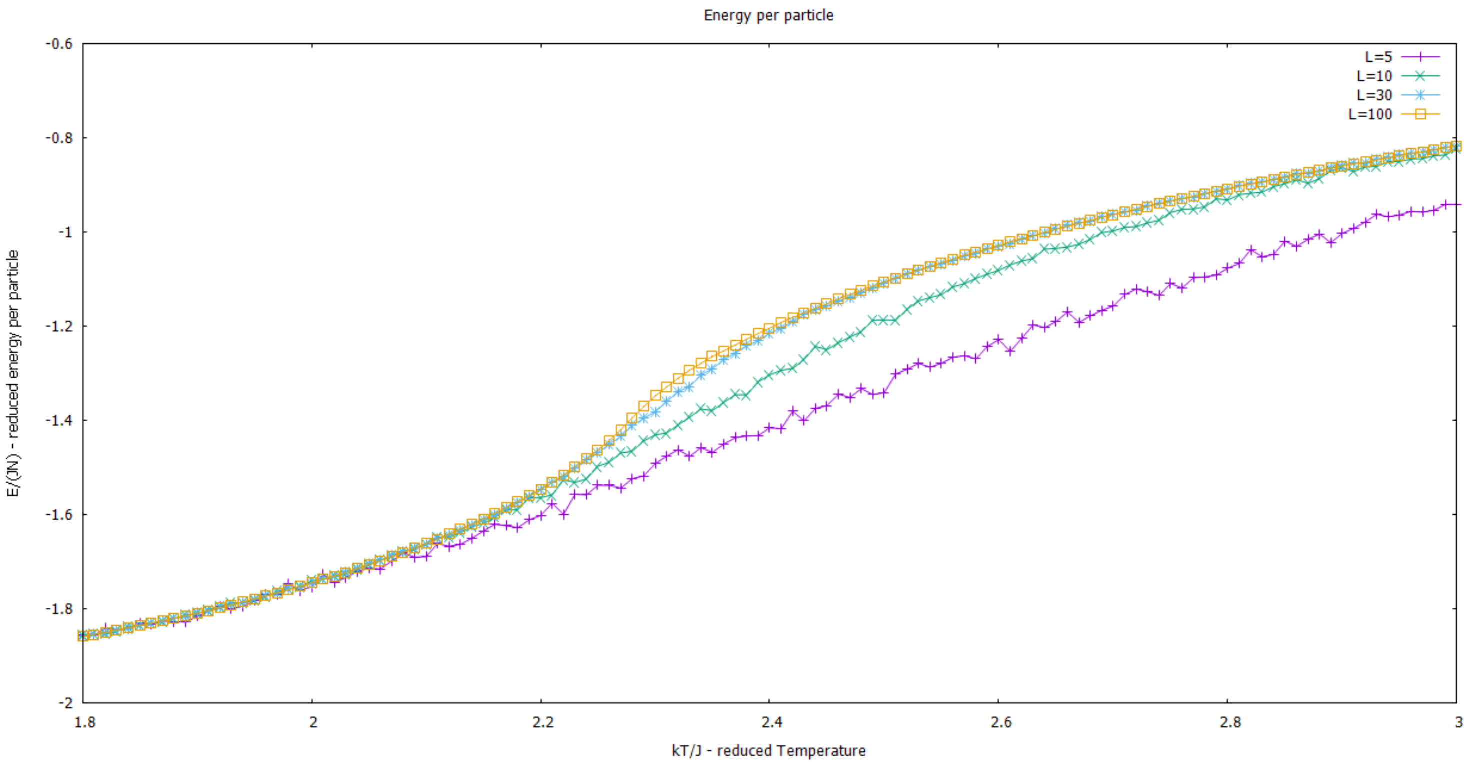
Z programu Skalowanie otrzymujemy zredukowaną temperaturę krytyczną, która wyniosła:

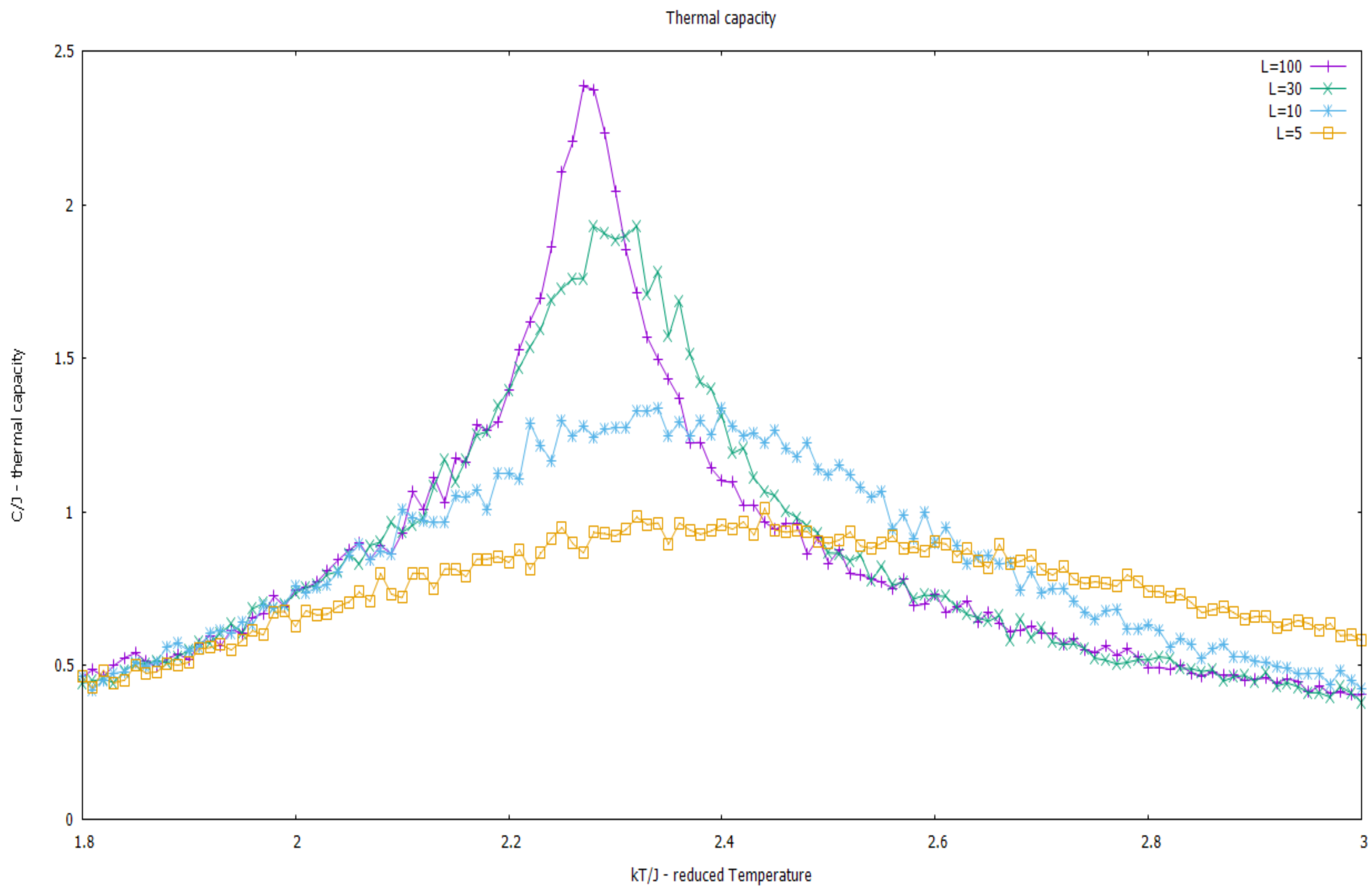
$$T_c^* = 2.2649994 \cdot$$

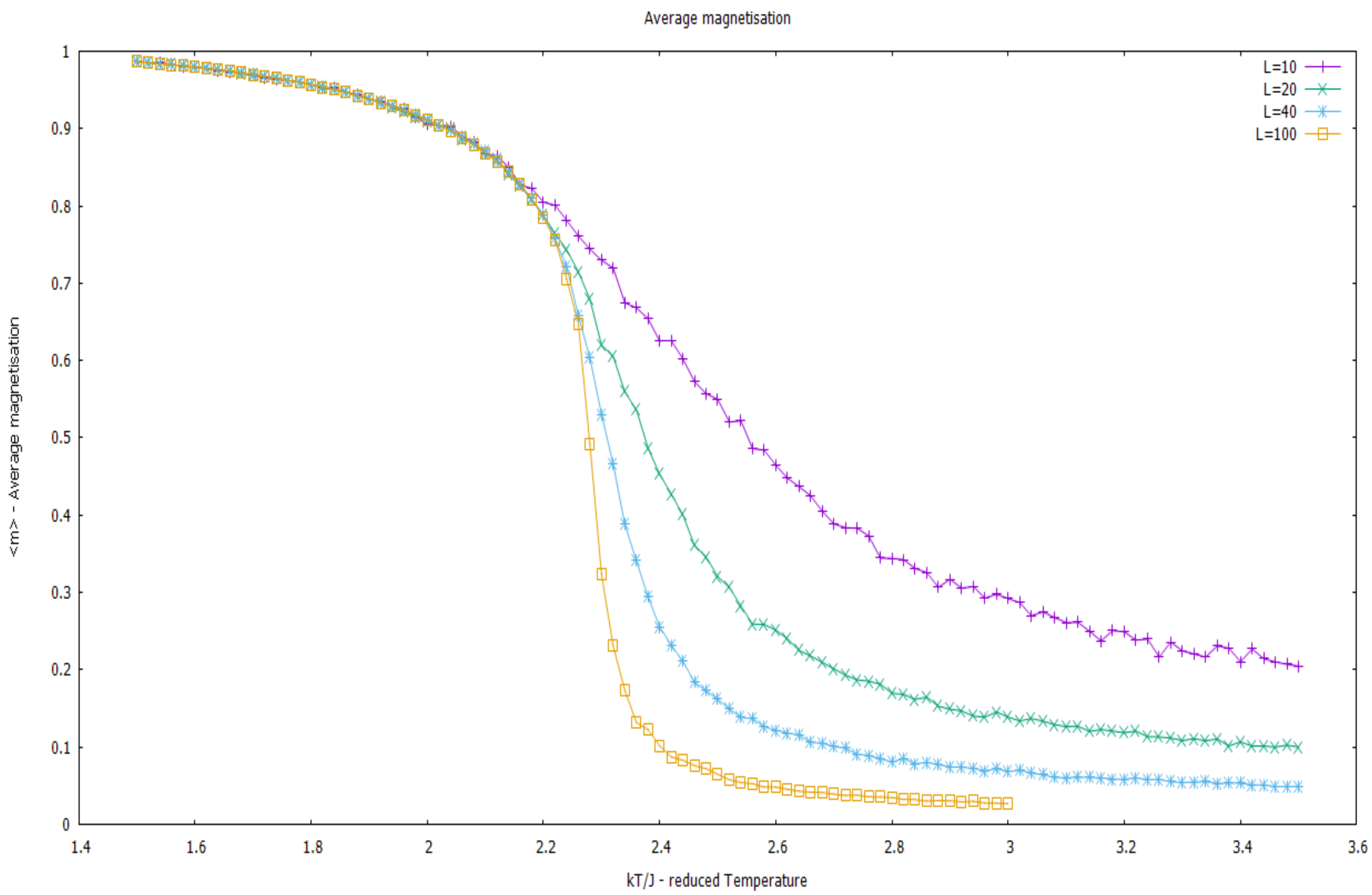
Lub używając zwykłej temperatury:

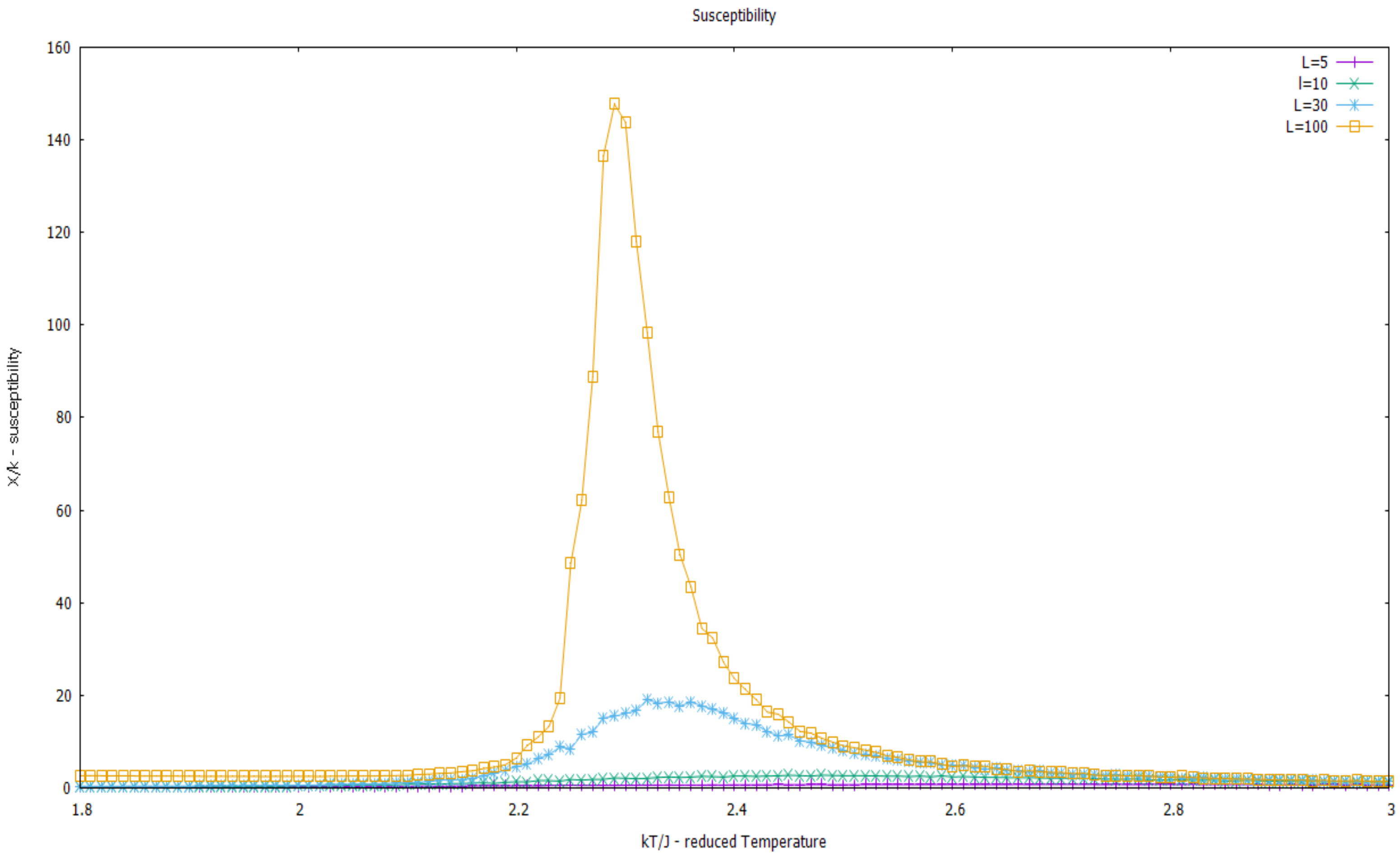
$$T_c = 2.2649994 \cdot \frac{J}{k}$$



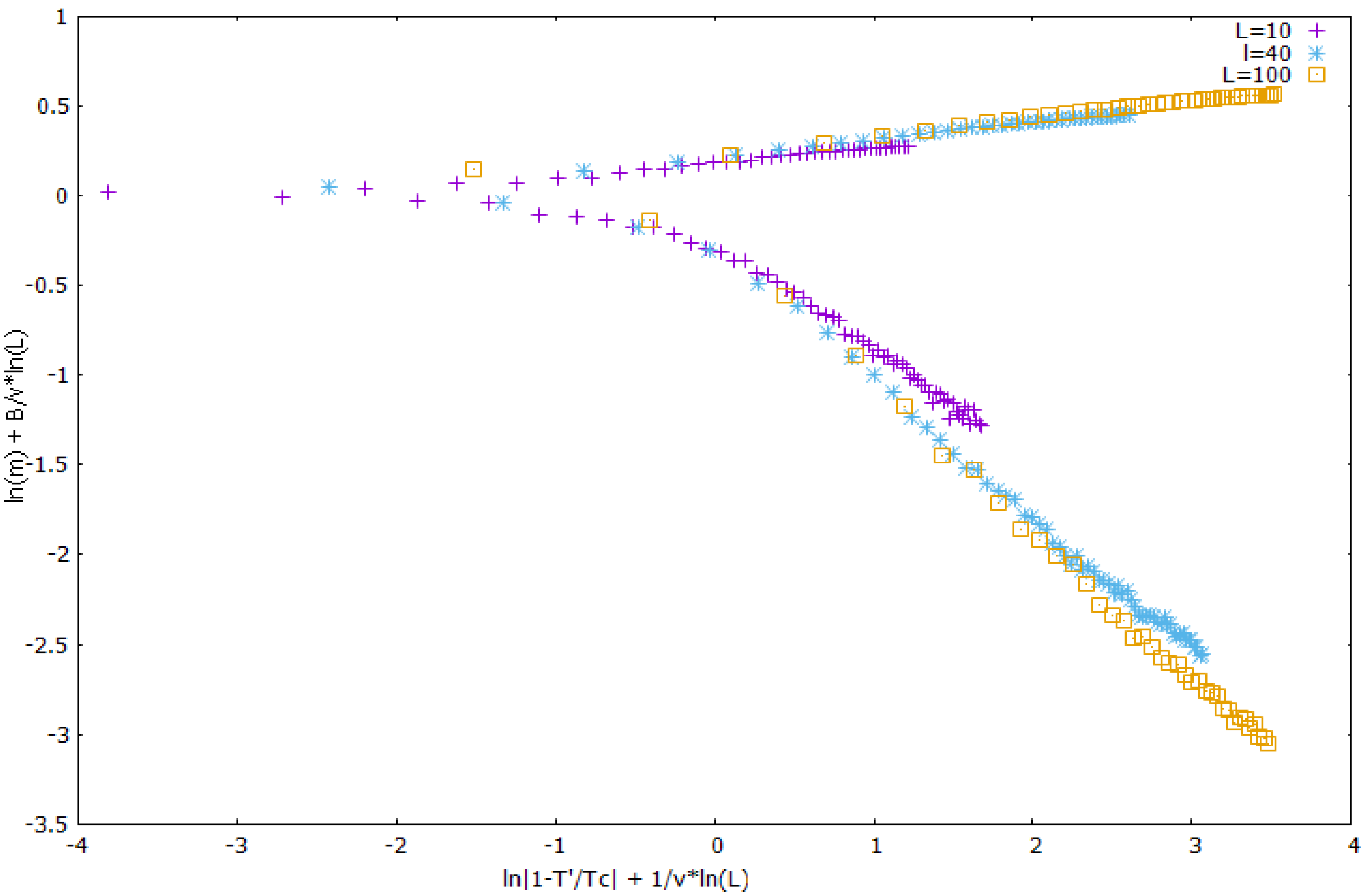



Thermal capacity
C/J - thermal capacity
kT/J - reduced Temperature
L=100
L=30
L=10
L=5
0
0.5
1
1.5
2
2.5
1.8
2
2.2
2.4
2.6
2.8
3


Average magnetisation
<m> - Average magnetisation
kT/J - reduced Temperature
L=10
L=20
L=40
L=100
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.4
1.6
1.8
2
2.2
2.4
2.6
2.8
3
3.2
3.4
3.6



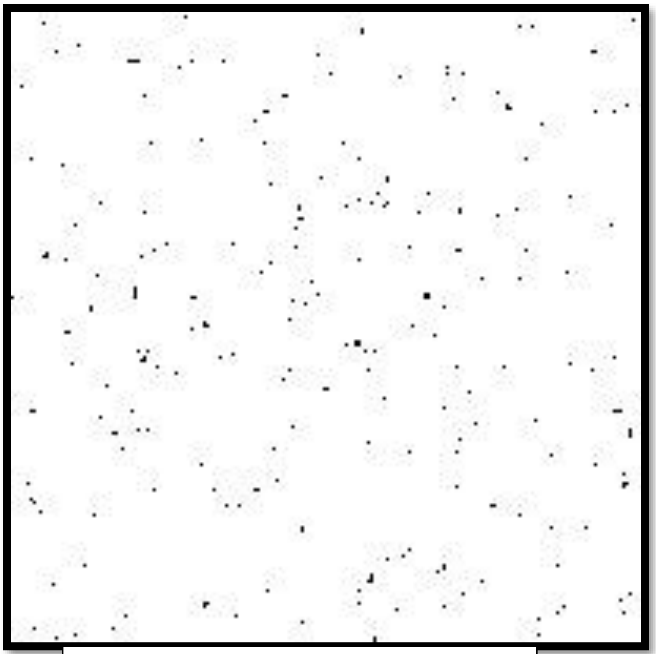
Susceptibility
X/k - susceptibility
kT/J - reduced Temperature
L=5
l=10
L=30
L=100



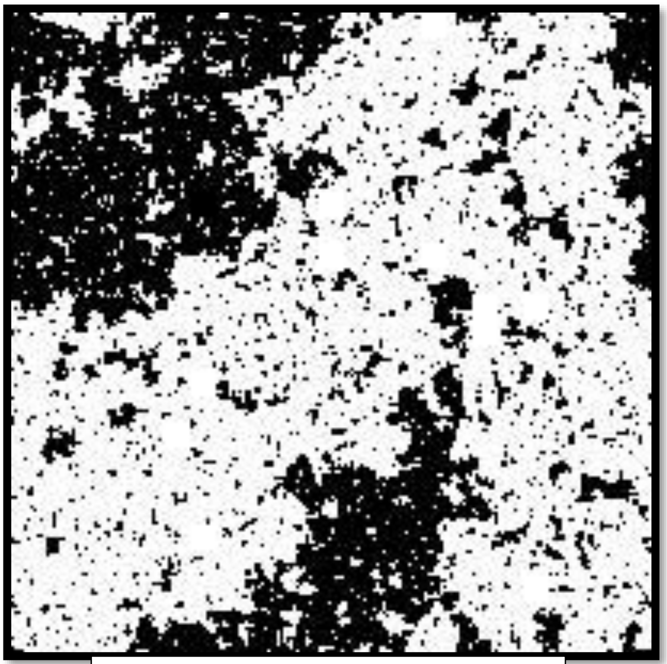
ln(m) + B/v*ln(L)
ln|1-T'/Tc| + 1/v*ln(L)
L=10
l=40
L=100

Model Isinga – Metoda Monte Carlo

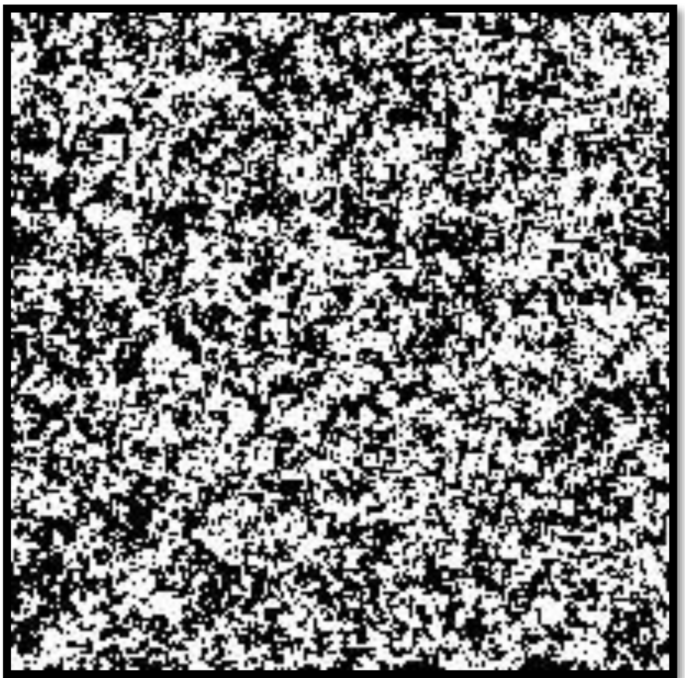
Poniżej przedstawiono układ 200x200 po MCS=500000 dla różnych temperatur, przy czym na czarno mamy spin s=-1, a na biało spin s=1:



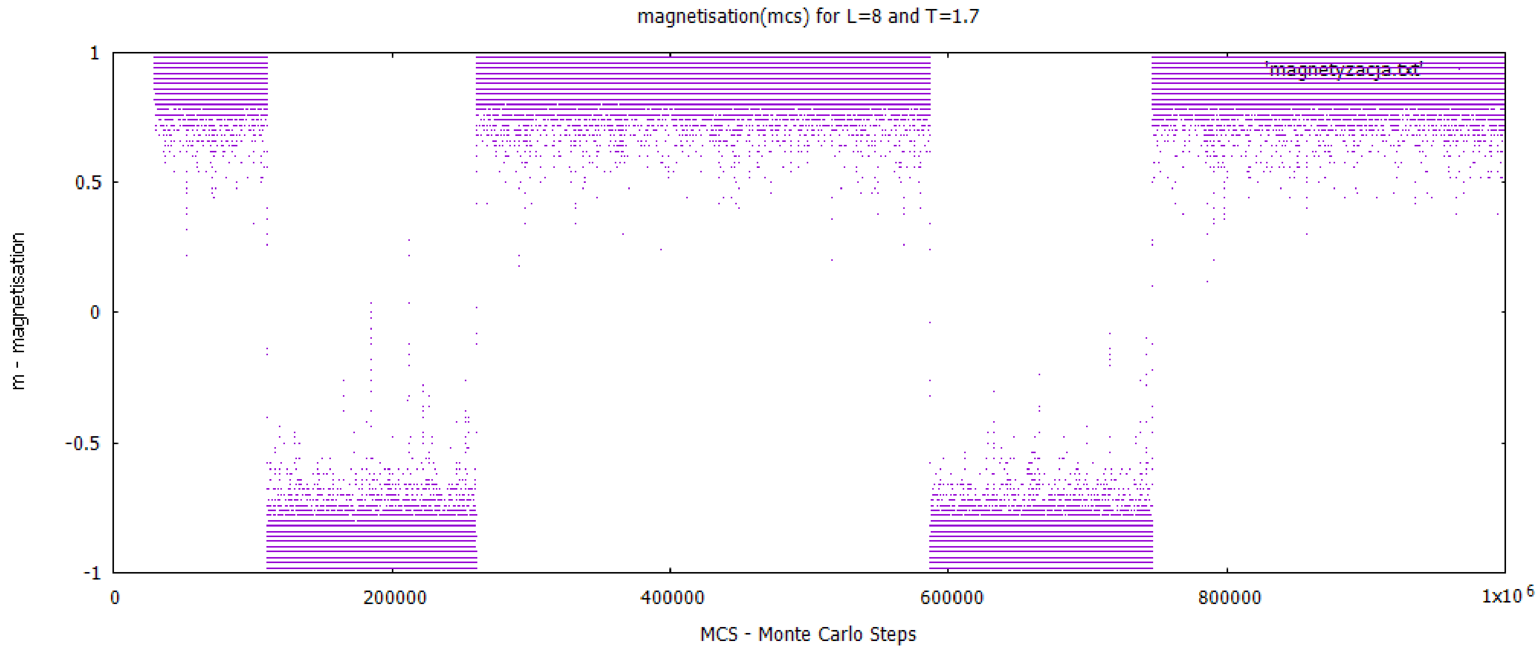
$T^* = 1.5 \quad \rightarrow \quad T = 1.5\frac{J}{k}$

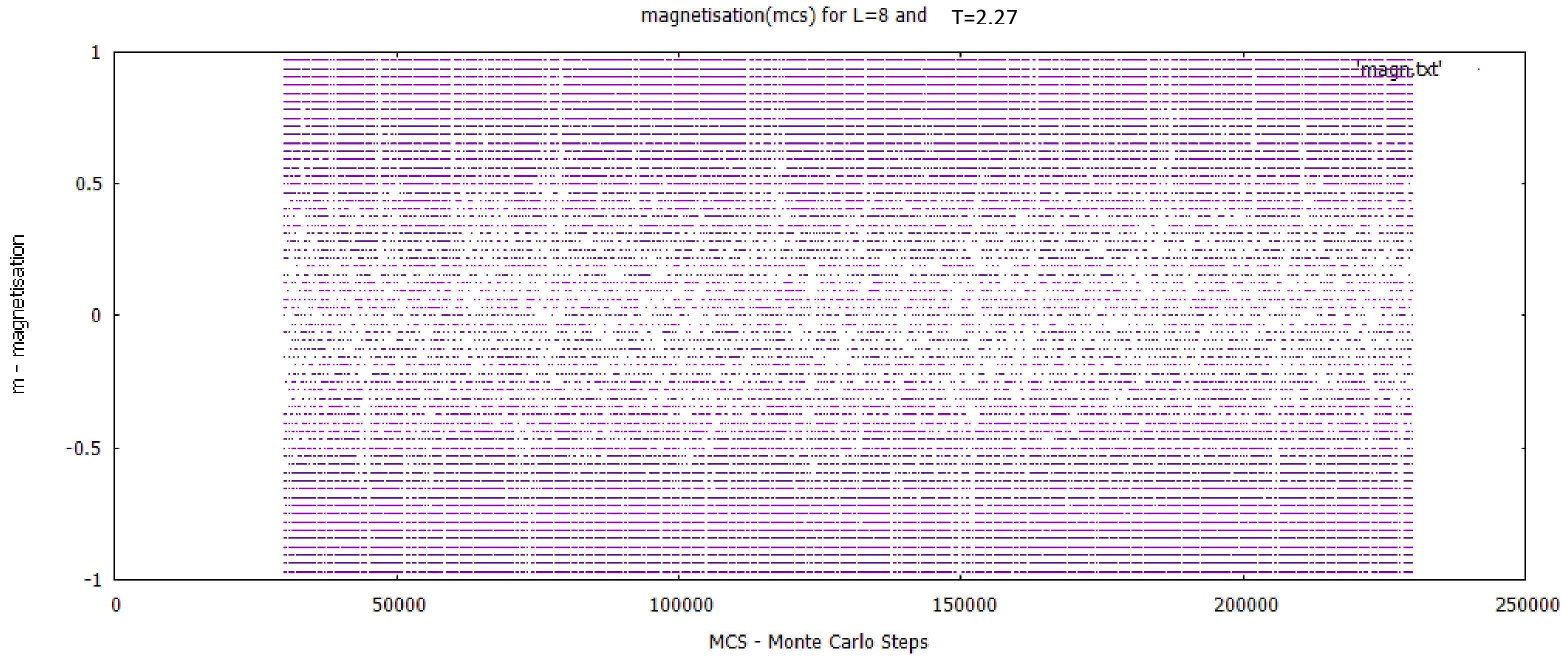


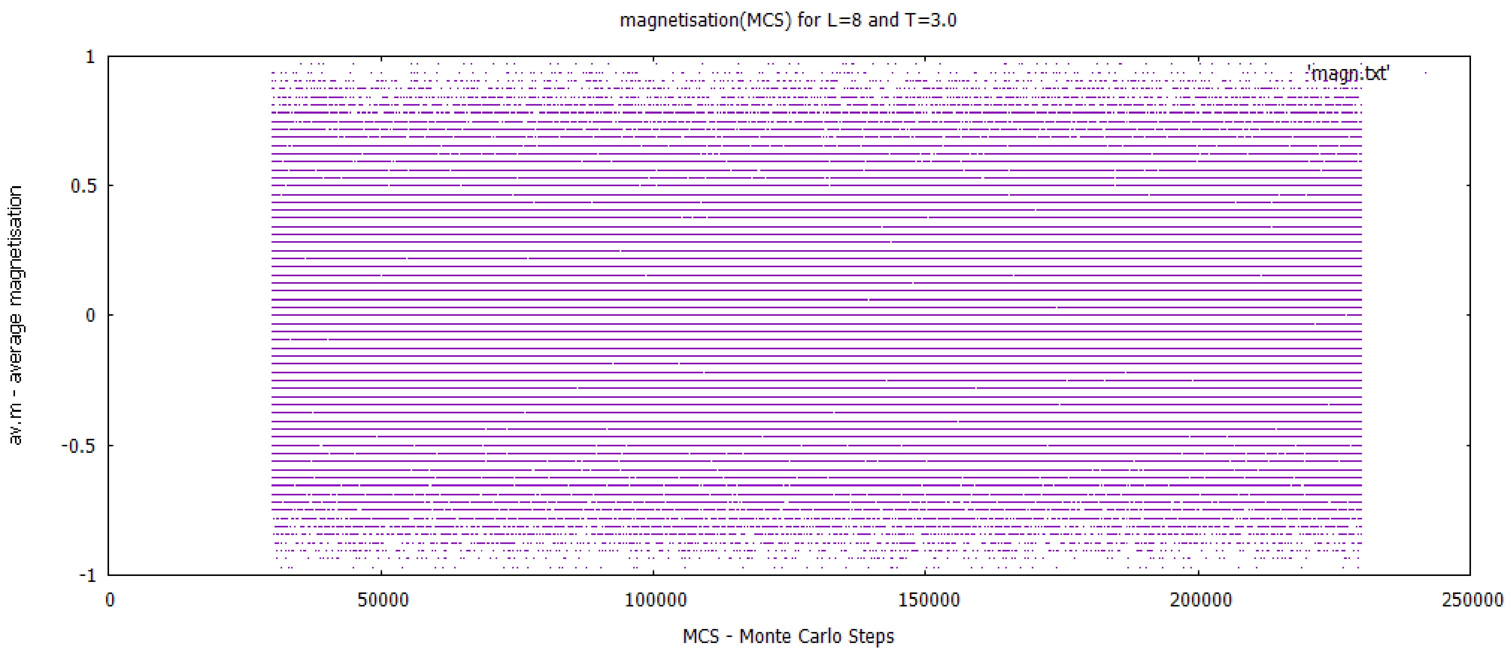
$T^* = 2.26 \quad \rightarrow \quad T = 2.26\frac{J}{k}$



$T^* = 3.3 \quad \rightarrow \quad T = 3.3\frac{J}{k}$


magnetisation(mcs) for L=8 and T=1.7
'magnetyzacja.txt'
m - magnetisation
MCS - Monte Carlo Steps
1
0.5
0
-0.5
-1
0
200000
400000
600000
800000
1x10 6


magnetisation(mcs) for L=8 and T=2.27
'magn.txt'
m - magnetisation
MCS - Monte Carlo Steps
1
0.5
0
-0.5
-1
0
50000
100000
150000
200000
250000


magnetisation(MCS) for L=8 and T=3.0
'magn.txt'
av.m - average magnetisation
MCS - Monte Carlo Steps
1
0.5
0
-0.5
-1
0
50000
100000
150000
200000
250000