Task 5 Ising Model

Mustapha Bousakla

December 3, 2021

Problem 66

Consider a 2D Ising model at a temperature $K = J/kT = 0.45$ and system size $L = 10$. Generate representative equilibrium configurations using the Metropolis algorithm with (i) random selection of the spin to update, (ii) a sequential selection and (iii) a check-board type update. Compute in every case the magnetization and check that the three methods give the same value within errors (make the error as small as possible within your computer capabilities).Repeat using the heat-bath algorithm for the generation of the configurations.

Our aim is to compute the mean value of the magnetization of the 2D Ising Model, whose Hamiltonian in the absence of an external magnetic field is:

$$H = -J \sum_{\langle i,j \rangle} s_i s_j \tag{1}$$

The mean value of the magnetization for every configuration of the spins is:

$$\langle m \rangle = m = \left| \frac{\sum_{i=1}^{N} s_i}{N} \right| \tag{2}$$

Where $N = L^2$ is the total number of spins. We will repeat this measure $M = 10^4$ times and hence the mean value of magnetization m will be:

$$m = \frac{\sum_{i=1}^{M} m_i}{M} \tag{3}$$

In the Metropolis algorithm we will randomly change the spins by means of three different methods and the new spin configuration will be accepted with probability $\min(1, e^{-\Delta H}) = \min(1, e^{-2KB_i})$. B_i depends on the changed spin i and its 4 nearest neighbors $s_{i\mu}$ and it is given by $B_i = s_i \sum_{\mu=1}^{4} s_{i\mu}$

Let's explain with more detail the three updating methods used in both Metropolis and heat-bath algorithms. In the first method included in Tables 1 and 2 we simply select randomly a spin i and we change its sign. If the change in energy ΔH is negative, it will be automatically accepted; otherwise, it will be accepted with probability $e^{-\Delta H}$: that is, a uniform random number between 0 and 1 u will be generated and the change in the spin i will be accepted if $u < e^{-2KB_i}$. We make this measure $M = 10^4$ times but with 2 important remarks: i) before the first measure we let the system thermalize with $M_{\text{therm}} = 100$ random spin changes; ii) the M measures are taken with $mc = 100$ iterations apart to avoid taking

too much correlated measures (that is, we make 100 acceptance/rejection spin changes before taking another measure). These M_{therm} and mc where enough to get consistent results with the other methods as it will be shown in tables 1 and 2.

In the second method the change in spins are made sequentially, and since I define the lattice as a matrix these changes are made row by row. In the third method we change the spins in a chessboard-like way: we first change the "black" positions and then the "white" ones. The changes are site by site, we do not change a whole row/column at once and neither do we in the previous method because in this way the acceptance probabilities would be low (small changes in H are more likely to be accepted).

Table 2 shows the results for the heat-bath method, in which the change of spin is always accepted. In the code I impose that a change $s_i \to s_i' = +1$ has a proposal probability given by:

$$g(s_i' = +1) = \frac{1}{1 + 2e^{a_i}} \tag{4}$$

where $a_i = K \sum_{i\mu} s_{i\mu}$. Afterwards we just generate a uniform number u and if $u < g(s_i' = +1)$, we set $s_i = +1$ and $s_i = -1$ otherwise.

Let's comment now the results in the tables. The errors in $\langle m \rangle$ take into account the correlation between the measures, which comes from the fact that the spin changes are not always accepted. The error in this case is:

$$\epsilon_m = \frac{\hat{\sigma}_m[\hat{m}]}{\sqrt{M}} \sqrt{2\tau + 1} \tag{5}$$

where $\sigma_m[\hat{m}]$ is the usual standard deviation of the measures:

$$\sigma_m^2[\hat{m}] = \frac{\sum_{i=1}^{M} m_i^2}{M} - \left(\frac{\sum_{i=1}^{M} m_i}{N} \right)^2 \tag{6}$$

and τ is the correlation time, approximated as:

$$\tau = \frac{\rho(1)}{1 - \rho(1)}; \qquad \rho(1) = \frac{\langle m_i m_{i+1} \rangle - \langle m_i \rangle \langle m_{i+1} \rangle}{\sigma_m[\hat{m}]} \tag{7}$$

Thus, the correlation time is approximated considering the correlation between two consecutive measures $\rho(1)$. The code for this is shown at the end (I decided to show at least one of the 6 Python codes to give more details about how the neighboring sites and τ are computed).

The first thing we note is that the values of the magnetization are coherent between the three updating methods and between both algorithms (Metropolis or heat-bath). Thus, $M = 10^4$ was large enough to get errors that reconcile between the total 6 measures. This sufficiently high value of M enabled short computational times as well, between 85 and 110 seconds. Had I used an M of one order of magnitude bigger I would have got errors of the order 10^{-4} though. We can also deduce the impact of the correlation time τ over the standard error $\sigma_m[\hat{m}]$ (equation 6): when τ is approximately 14, the error for correlated measures becomes approximately $\sqrt{2\tau + 1} \approx 5$ times bigger than the error if the measures were uncorrelated (equation 6). In the random spin flip updating method is where τ is bigger and it could have been reduced by increasing the time/iterations (mc) between measures, which in turn dramatically increases the computational time. In Tables 1 and 2 both errors (the correlated and uncorrelated) are shown to prove the influence of the correlation time in the real error.

Metropolis Method	$\langle m \rangle$	τ	$\hat{\sigma}_m[\hat{m}]/\sqrt{M}$
Random spin selection	0.7919 ± 0.0085	10.65	0.0018
Sequential update	0.7985 ± 0.0049	3.46	0.0017
Checkboard update	0.7927 ± 0.0066	6.04	0.0018

Table 1: Mean value of the magnetization by three different updates methods using Metropolis algorithm. The error in $\langle m \rangle$ already includes the correlation time τ as in (5).

Heat-bath Method	$\langle m \rangle$	τ	$\hat{\sigma}_m[\hat{m}]/\sqrt{M}$
Random spin selection	0.796 ± 0.010	14.45	0.0018
Sequential update	0.7918 ± 0.0074	7.73	0.0018
Checkboard update	0.8000 ± 0.0060	5.46	0.0017

Table 2: Mean value of the magnetization by three different updates methods using the heat-bath algorithm.The error in $\langle m \rangle$ already includes the correlation time τ as in (5).

#METROPOLIS RANDOM SPIN SELECTION

```
import numpy as np
import math
import itertools
import random
import time


time_start = time.clock()
L=10
N=L**2
k=0.45
mc=100
M=10000
Mtherm=100

s=np.random.choice([1, -1], size=(L, L))

def neighbors(s, L, x, y):
    left   = (x, y - 1)
    right  = (x, (y + 1) % L)
    top    = (x - 1, y)
    bottom = ((x + 1) % L, y)

    return [s[left[0], left[1]],
            s[right[0], right[1]],
            s[top[0], top[1]],
```

```
            s[bottom[0], bottom[1]]]



for q in range(Mtherm):
    i=np.random.randint(0, L)
    j=np.random.randint(0, L)
    ib=s[i][j]*np.sum(neighbors(s,L,i,j))
    if np.random.uniform()< np.exp(-2*k*ib):
        s[i][j]=-s[i][j]



rm1=np.float(abs(sum(s[p][q] for p in range(L) for q in range(L))))/N
c=0
rm2=0
rm=0
for u in range(M):
    for w in range(mc):          #we measure every N updates of the spins
        i=np.random.randint(0, L)
        j=np.random.randint(0, L)
        ib=s[i][j]*np.sum(neighbors(s,L,i,j))
        if ib<=0:
            s[i][j]=-s[i][j]
        if ib>0 and np.random.uniform()< np.exp(-2*k*ib):
            s[i][j]=-s[i][j]

    rm0=np.float(abs(sum(s[p][q] for p in range(L) for q in range(L))))/N
    rm=rm+rm0
    rm2=rm2+rm0**2
    c=c+rm0*rm1
    rm1=rm0



mag=np.float(rm)/M
rm2=np.float(rm2)/M-mag**2
c=np.float(float(c)/M-mag**2)/rm2
tau=np.float(c)/(1-c)
errorsintau=np.sqrt(float(rm2)/M)
error=np.sqrt(float(rm2)*(2*tau+1)/M)

print(mag,tau,errorsintau,error)


time_elapsed = (time.clock() - time_start)
print(time_elapsed)
```