Ayush Paliwal
Matriculation Number: 21915793

A. 2D-Ising model using Metorpolis Algorithm -

Structure and Initialising Constants

- After giving 2D-Ising model its structure, i.e. a NxN lattice, where N=16. Initialising initial conditions with T=KB(Boltzmann)=1, and J(to vary)
- Starting with all spins up, state(0), energy to this state was initialised using the energy function described as $H/K_BT = -J\sum_{<ij>} s_i s_j + h\sum_i s_i$

```
def energy(s):
    term1=0
    term2=0

    for i in range(N):
        for j in range(N):
            if j!=N-1:
                term1+=s[i][j]*s[i][j+1]+s[i][j]*s[i][j-1]
        term1+=s[i][N-1]*s[i][N-2]+s[i][N-1]*s[i][0]



    for j in range(N):
        for i in range(N):
            if i!=N-1:
                term2+=s[i][j]*s[i+1][j]+s[i][j]*s[i-1][j]
        term2+= s[N-1][j]*s[N-2][j]+s[N-1][j]*s[0][j]
    return(-Jc*(term1+term2)/2);
```

Fig1. Shows the function to assign energy to the states and application of periodic boundary conditions.

•Periodic Boundary Conditions are applied.

•With h being initialised to 0, we only have the contribution from the coupling factor.

Flipping the spin and Acceptance Probability

- (Step 1) Two random numbers (r1 and r2) are generated, which are normalised to lie between the lattice dimensions i.e. $0 \leq r1, r2 < N$.
- These random numbers describe the point in lattice where spin is to be flipped (r1,r2).

```
test[r1,r2]*=-1
```

- Probability π(r1,r2)=$1/N^2$
- We accept the spin flip with π and check for the Acceptance Probability of the new state. The Acceptance Probability (p)- $\alpha_M(\{s'\}\,|\,\{s\}) = metrop(e^{-\Delta H/K_BT})$
- We generate another random number between 0 and 1, and compare p with the it.

```
if p1>=random.random():
```

If p≥1
- We accept the new state, and go to step 1 again.

Otherwise, we reject this new state and let our system exist in previous state for the current cycle, and go to step 1 again.

- For faster working of code (Fig.2), I calculate the energy change occurring locally due to the spin flip i.e. near the nearest neighbours of the point where spin is flipped.

```
def diffE(test,r1,r2):
    horiz=0
    verti=0
    if r2!=N-1:
        horiz+=test[r1,r2]*2*(test[r1][r2-1]+test[r1][r2+1])
    else:
        horiz+=test[r1,r2]*2*(test[r1][r2-1]+test[r1][0])
    if r1!=N-1:
        verti+=test[r1,r2]*2*(test[r1-1][r2]+test[r1+1][r2])
    else:
        verti+=test[r1,r2]*2*(test[r1-1][r2]+test[0][r2])

    return (-Jc*(horiz+verti))
```

Fig.2

Physical Intuition of the Program -
With, every spin flip, we describe a new state and check for its energy. System wants to be in the lowest energy state possible. So all those transitions which have lower energy than the previous state, are always accepted.
To ensure proper treatment of entropic contributions, i.e. to maximise the free energy, we still have to take in account of p being less than 1 and still being accepted as the new state.

B. Verification of the Program

- **Energy Check :**

We compare the net energy changes with the difference in energies of the initial and the final state.

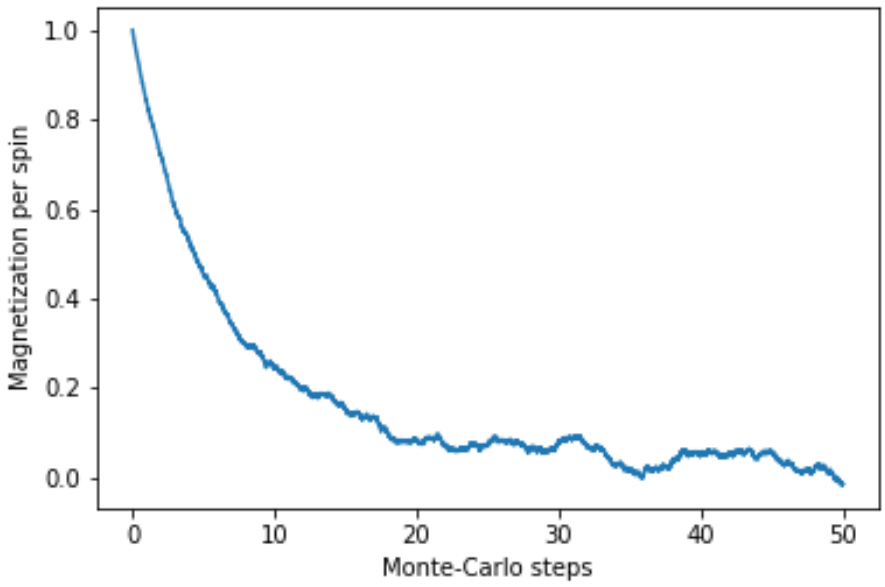
```
energydstore[i]=diffE(test,r1,r2)
  ,energy(state[:,:,-1])-iniE)
```

```
In [74]: runfile('/Users/ayush/Desktop/pp.py', wdir='/Users/ayush/Desktop')
Net Energy Change = 88.13735870195423, Energy difference between initial and final
state = 88.1373587019543
```

- **Single Histogram Reweighing:**

C. The **relaxation of the magnetization per spin**, averaged over **50 realizations** of the simulation runs at **50 MCMC steps** after starting from the configuration with m = +1. Plot shows magnetisation per spin versus the number of Monte-Carlo steps for J = 0.3. A single MCMC step consisted of N^2 (256 here) runs. Fig.3.

Fig3. $[m]_s$ vs MC steps



F. **Non-Boltzmann Sampling - Wang Landau Algorithm**

- Making storage for book-keeping of magnetisation and weight functions.

```
wmagstore=np.ones(steps)
wts=np.ones(N**2+1)#initialising all weights to 1
```

- An additional weighing parameter/function is fed to the equilibrium probability distribution of every state possible.

- Our metropolis acceptance probability changes to -
$\alpha_M(\{s'\}\,|\,\{s\}) = metrop[e^{-\Delta H/K_BT} \times (w(m\{s\})/w(m\{s'\}))]$.

- Initialising to every state, weight w({s})=1. Keeping the modification factor f=1.2 in the start.

- Each time we hit a state {s} with magnetisation m, we increase the weight. $w(M) \rightarrow f\,w(m)$. and the histogram or the occurrence of the magnetisation is recorded for that state.

```
m=int(np.sum(s))
wmagstore[m]+=ii
wts[m]=np.log(f)+wts[m]
```

- When all the states are met equally (Check for flatness), we downgrade the modification factor to $(f)^{1/2}$ and re-initialise the magnetisation array to containing only ones.

```
hmax=np.max(wmagstore)
hmin=np.min(wmagstore)
if  (hmax-hmin)/(hmax+hmin)<0.2:
```

- The process is repeated till f reaches the order of $1+10^{-6}$.

```
if f==1+10e-6:
    flag=1

while flag==0:
```

- At the end of this process, we have our weights corresponding to every state possible in the system.

G.

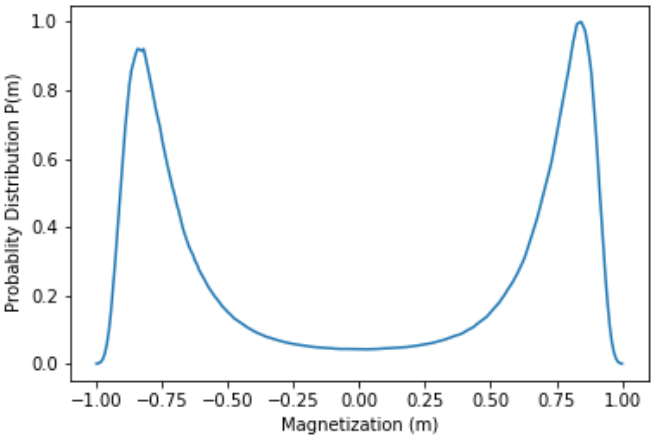


Fig.3 Probability distribution as graphed using Wang Landau Sampling

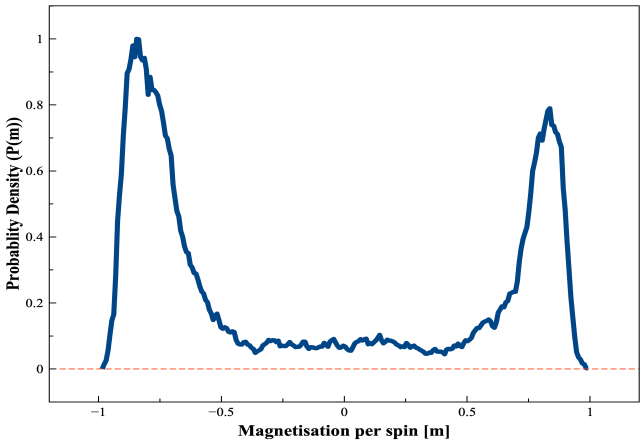


Fig.4 Probability distribution as graphed using the importance sampling in MCMC process.

Fig3. is the Wang-Landau program run by calculating the entire probability distribution at the critical temperature $J = J_c = \ln(1+\sqrt{2})/2$ and comparing the result to that of part b Fig.4.

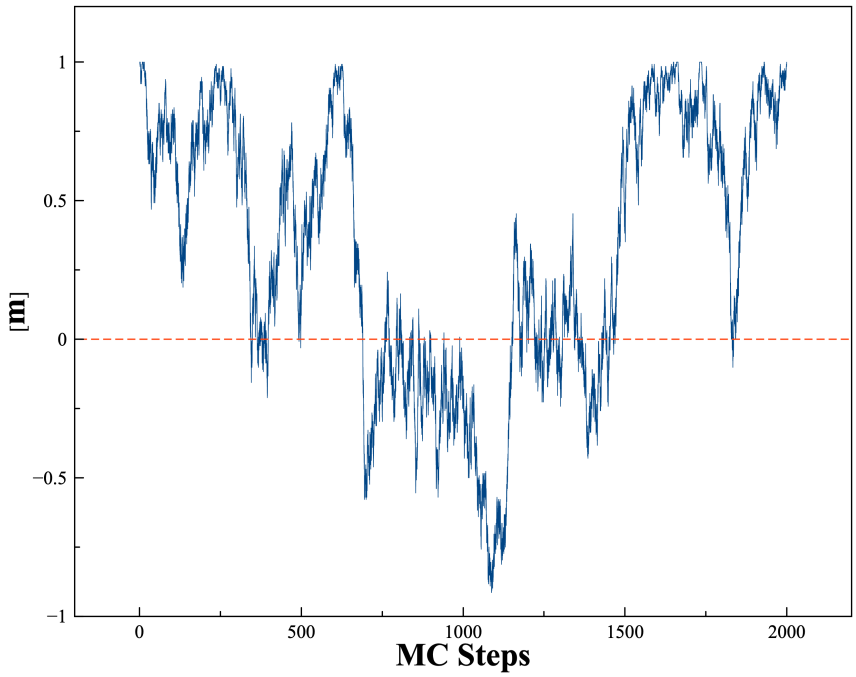


Fig.5 “Time” series of magnetisation over MC steps.

H. Wang Landau process for J=1

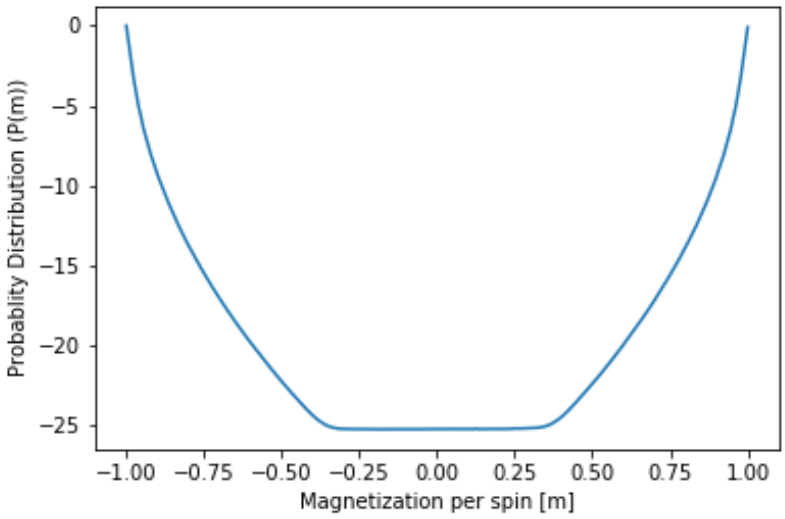


Fig.6 For J=1, calculated probability distribution of magnetisation per spin. y-axis is log(P(m)).

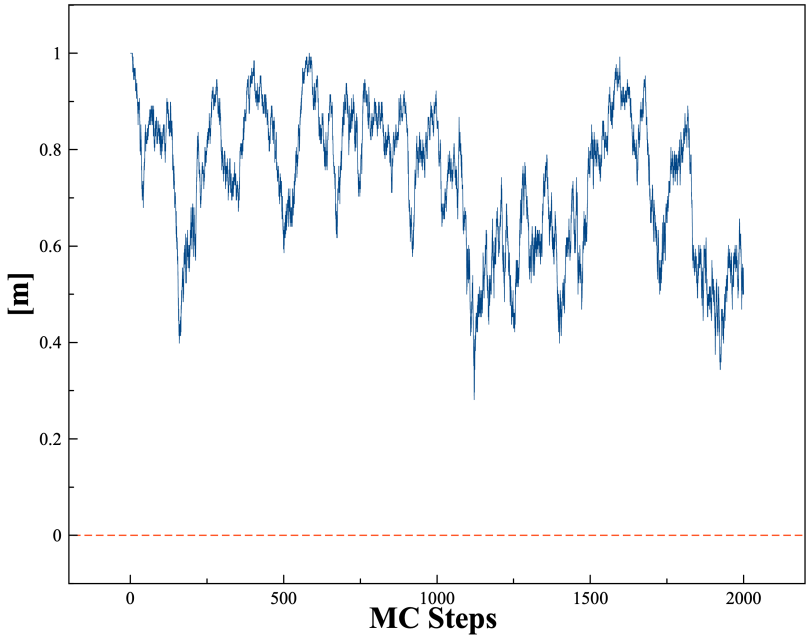


Fig.6 Time series of magnetisation for J=1 using Wang Landau sampling.