2D Monte Carlo Ising Model

Group 18

Ceran Hu
Bingran You
Sichen Yue

Berkeley
UNIVERSITY OF CALIFORNIA

1. Theory
2. Implementation
3. Results

Berkeley
UNIVERSITY OF CALIFORNIA

Theory

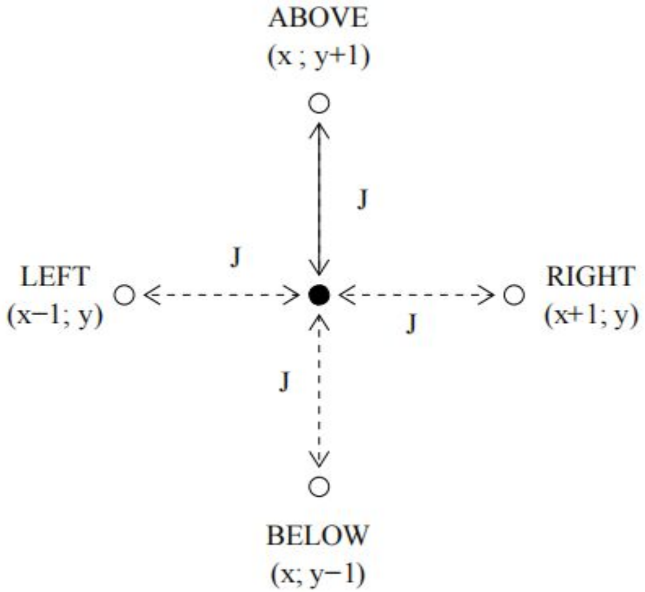
$$\langle A(x)\rangle_T = \frac{1}{Z}\int e^{-\beta H(x)}A(x)dx$$

$$Z = \int e^{-\beta H(x)}dx$$

$$P(x) = \frac{1}{Z}e^{-\beta H(x)}.$$

Theory

$$H_i = -J \sum_{j_{nn}} s_i s_j$$



Berkeley
UNIVERSITY OF CALIFORNIA

Theory

For a lattice of size (K,L), the 2D analytic solution for Ising Model:

$$k = \frac{1}{\sinh(2K)\sinh(2L)}$$

$$U = -J\coth(2\beta J)\left[1 + \frac{2}{\pi}(2\tanh^2(2\beta J) - 1)\int_0^{\pi/2} \frac{1}{\sqrt{1 - 4k(1+k)^{-2}\sin^2(\theta)}}d\theta\right]$$

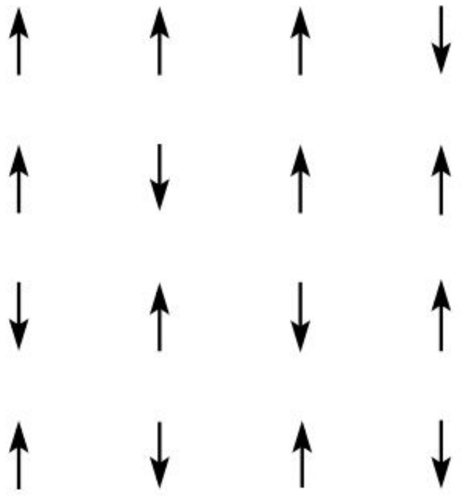
$$\frac{kT_c}{J} = \frac{2}{\ln(1+\sqrt{2})} \approx 2.26918531421$$

$$M = \left[1 - \sinh^{-4}(2\beta J)\right]^{1/8}$$

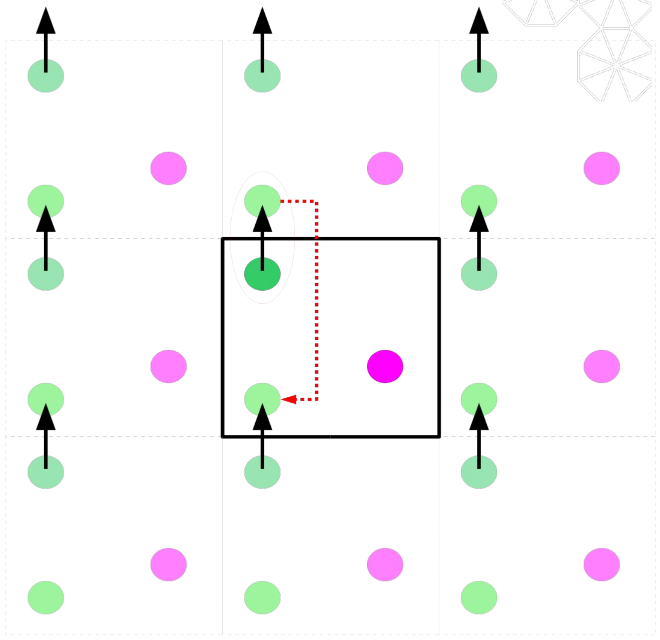
Berkeley
UNIVERSITY OF CALIFORNIA

Theory

1. Consider on a finite lattice



2. Periodic Boundary Condition



Theory

$$\langle M \rangle = \frac{1}{N} \sum_{\alpha}^{N} M(\alpha)$$

$$\langle E \rangle = \frac{1}{2} \langle \sum_{i}^{N} H_i \rangle = \frac{1}{2} \langle -J \sum_{i}^{N} \sum_{j_{nn}} s_i s_j \rangle$$

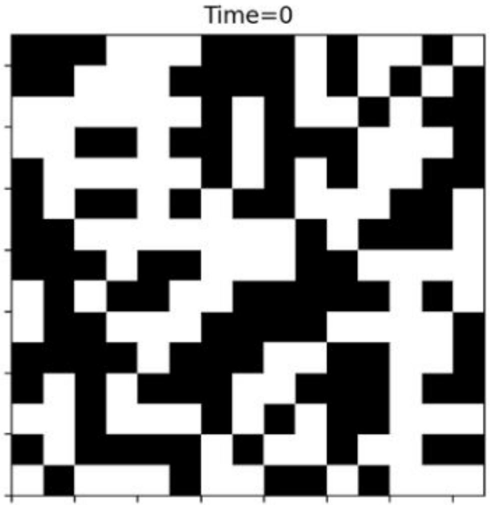
$$C = \frac{\partial E}{\partial T} = \frac{(\Delta E)^2}{k_b T} = \frac{\langle E^2 \rangle - \langle E \rangle^2}{k_b T^2}$$

$$\chi = \frac{\partial M}{\partial T} = \frac{(\Delta M)^2}{k_b T} = \frac{\langle M^2 \rangle - \langle M \rangle^2}{k_b T}$$

Berkeley
UNIVERSITY OF CALIFORNIA

Implementation

- 16*16 grid
- Initial grid randomly generated
- Setting iterations and temperature
- Only consider 4 nearest spins



Implementation

MC Simulation

$$\sigma'_k = -\sigma_k$$

$$p_{eq}(x_i) = \frac{1}{Z}\exp(-H(x_i)/k_B T)$$

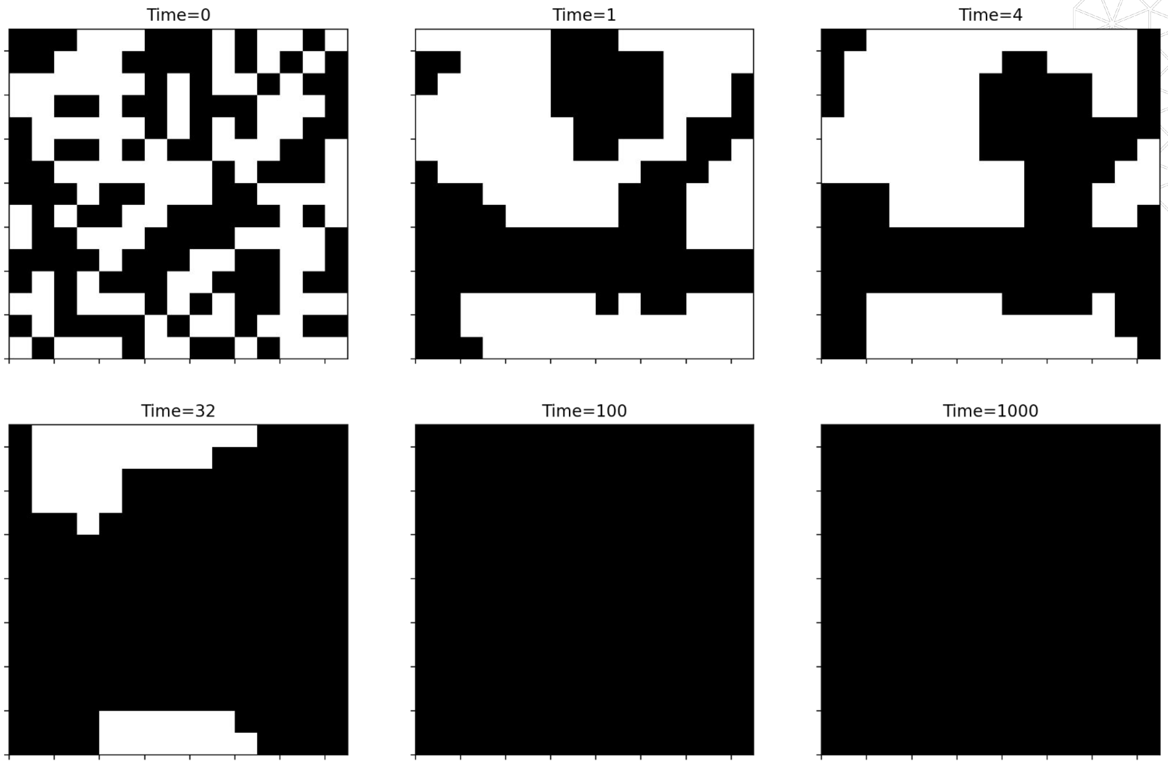
$$\frac{w(\sigma_{\mathrm{k}} \to \sigma'_{\mathrm{k}})}{w(\sigma'_{\mathrm{k}} \to \sigma_{\mathrm{k}})} = \exp\left(-\frac{\Delta E_k}{k_B T}\right)$$

```python
class Ising():
    ''' Simulating the Ising model '''

    ## monte carlo moves
    def mcmove(self, config, N, beta):
        ''' This is to execute the monte carlo moves using
        Metropolis algorithm such that detailed
        balance condition is satisified'''
        for i in range(N):
            for j in range(N):
                    a = np.random.randint(0, N)
                    b = np.random.randint(0, N)
                    s =  config[a, b]
                    nb = config[(a+1)%N,b] + config[a,(b+1)%N] + config[(a-1)%N,b] + config[a,(b-1)%N]
                    cost = 2*s*nb
                    if cost < 0:
     def simulate(self):
        ims = []
        size = (10, 10)
        #figure = plt.figure()
        FIG, ax = plt.subplots(figsize=size)

        '''Parameter initialization'''
        N = 16,
        temperature = .4
        iterations = 1001

        '''Simulation and data saving'''
        temp = temperature
        config = 2*np.random.randint(2, size=(N,N))-1
        msrmnt = iterations
        for i in range(msrmnt):
            self.mcmove(config, N, 1.0/temp)
            im = ax.imshow(config, 'gray');
            ims.append([im])
        writer = animation.PillowWriter()
        ani = animation.ArtistAnimation(FIG, ims, interval=50, blit=True, repeat_delay=1000)
        plt.show()
        ani.save('test.gif', writer=writer)
```

Berkeley
UNIVERSITY OF CALIFORNIA

Results

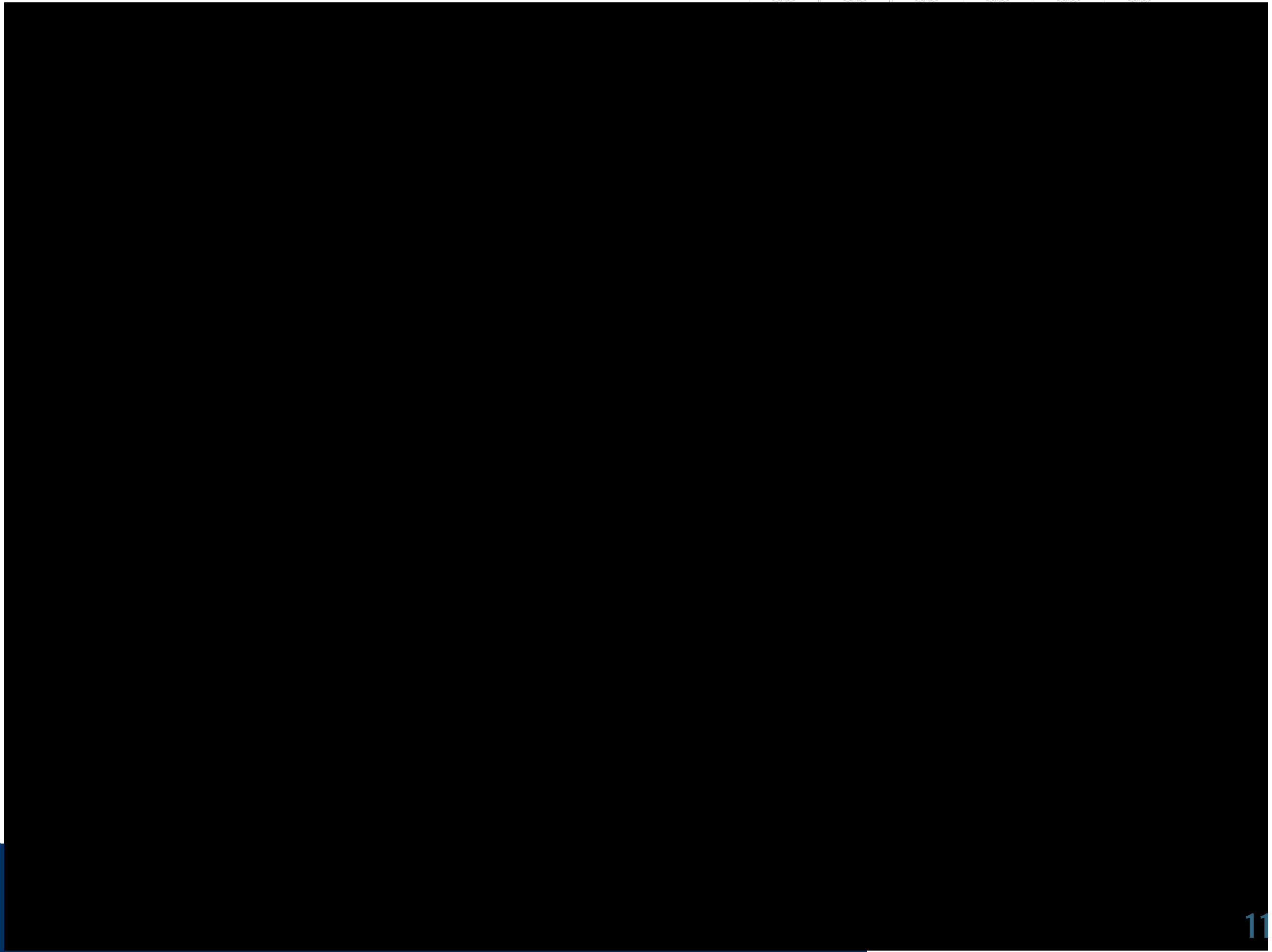
Snapshots of grids



Results

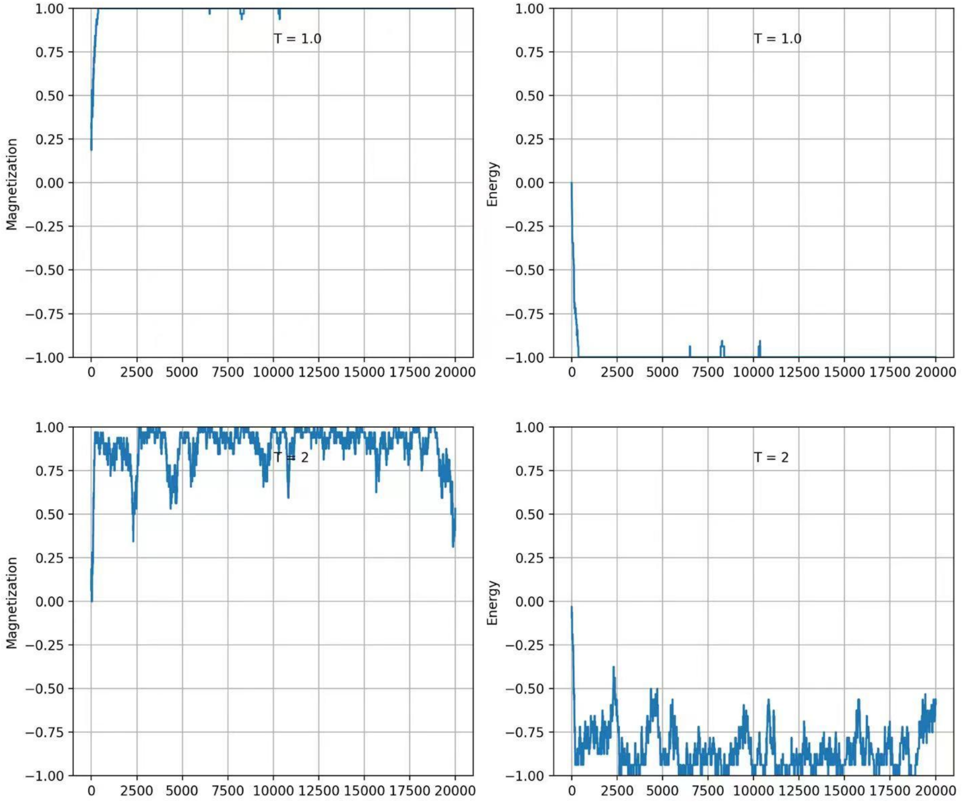
Animation

(Temperature = 0.4)



Results

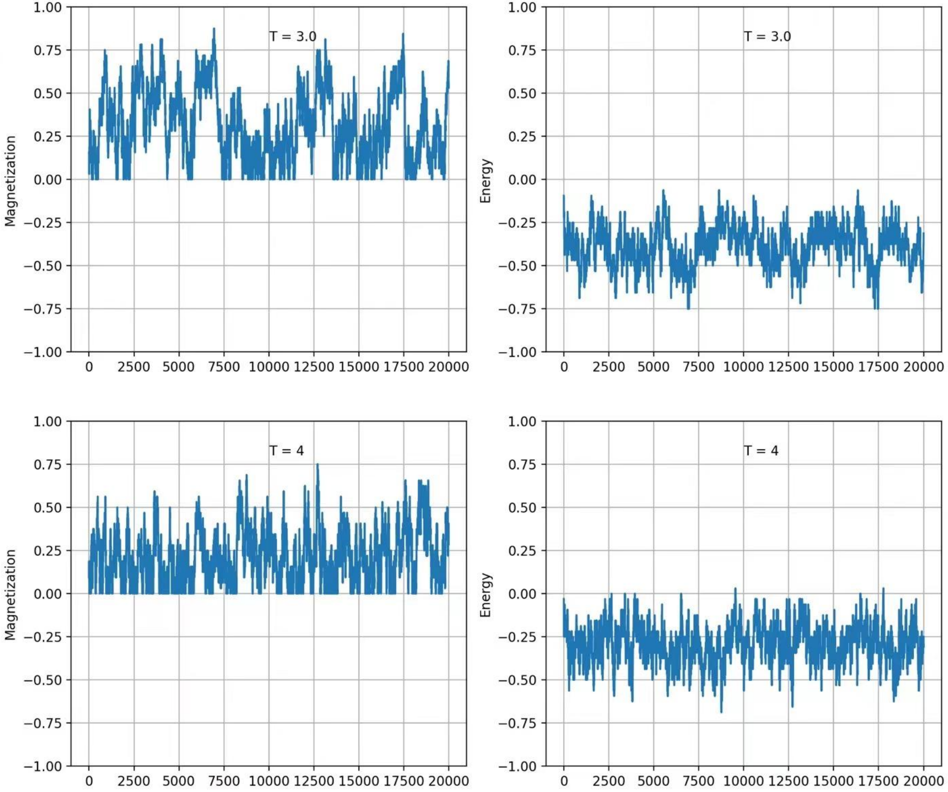
Energy & Magnetization



Berkeley
UNIVERSITY OF CALIFORNIA

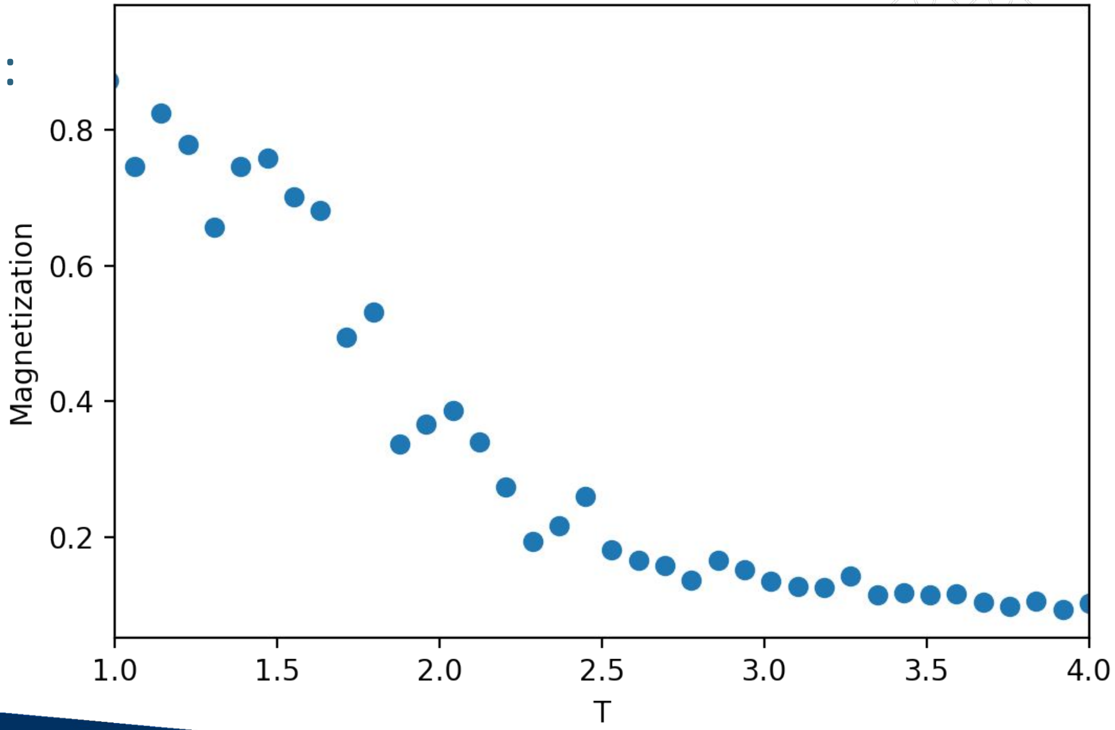
Results

Energy & Magnetization



Results Analysis

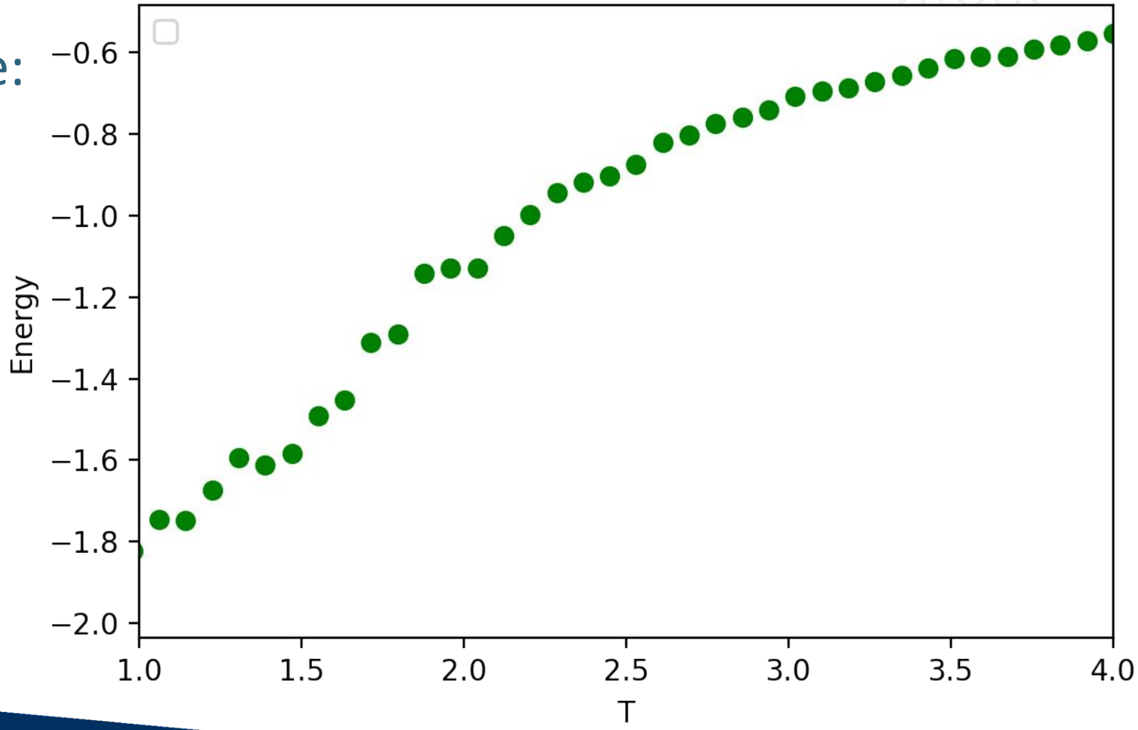
Variation with Temperature:



Berkeley
UNIVERSITY OF CALIFORNIA

Results Analysis

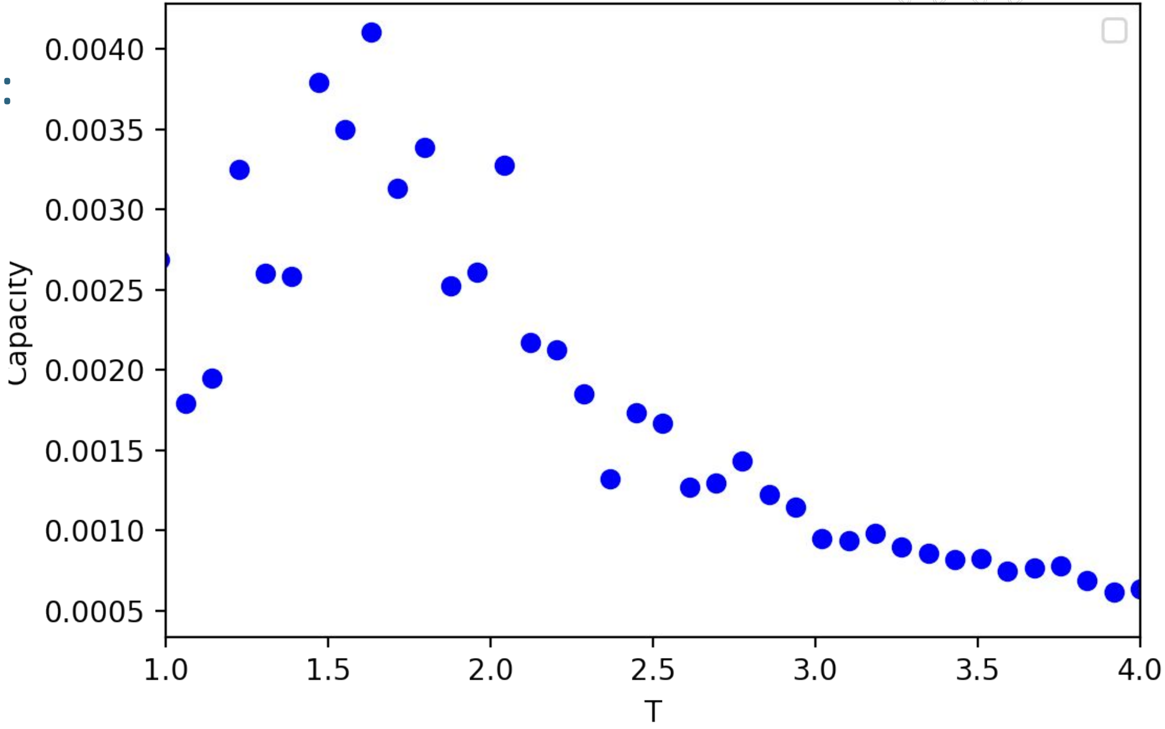
Variation with Temperature:



Berkeley
UNIVERSITY OF CALIFORNIA

Results Analysis

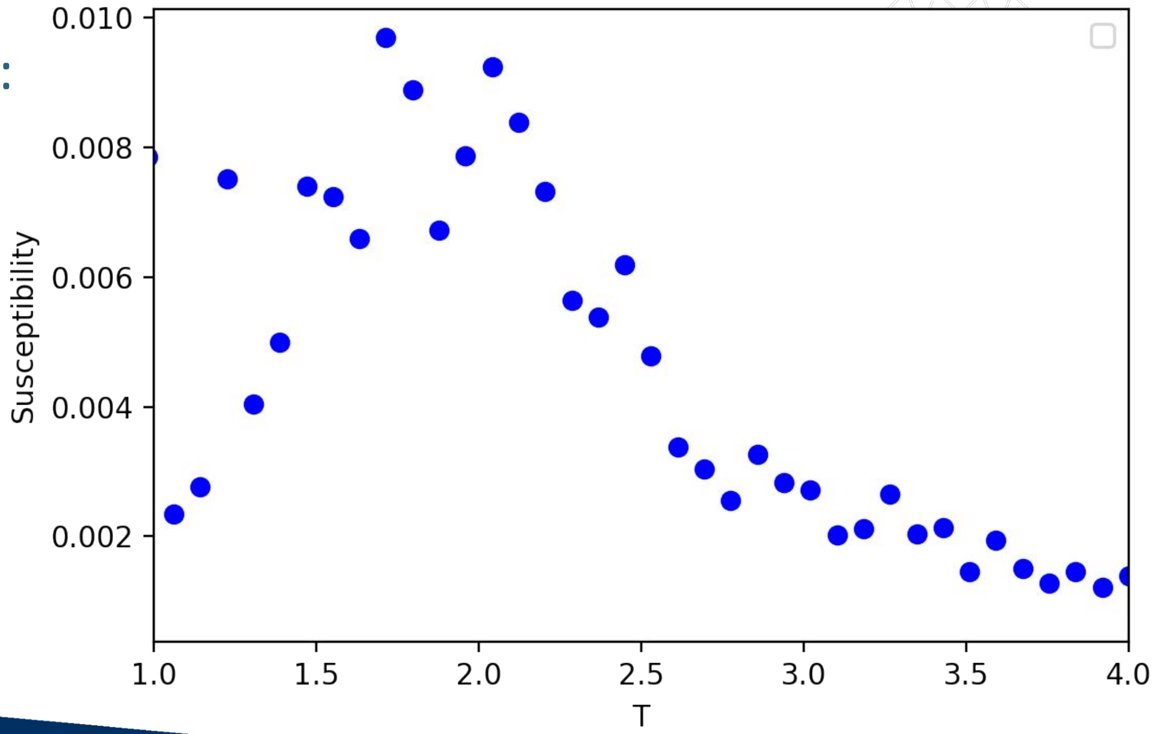
Variation with Temperature:



Berkeley
UNIVERSITY OF CALIFORNIA

Results Analysis

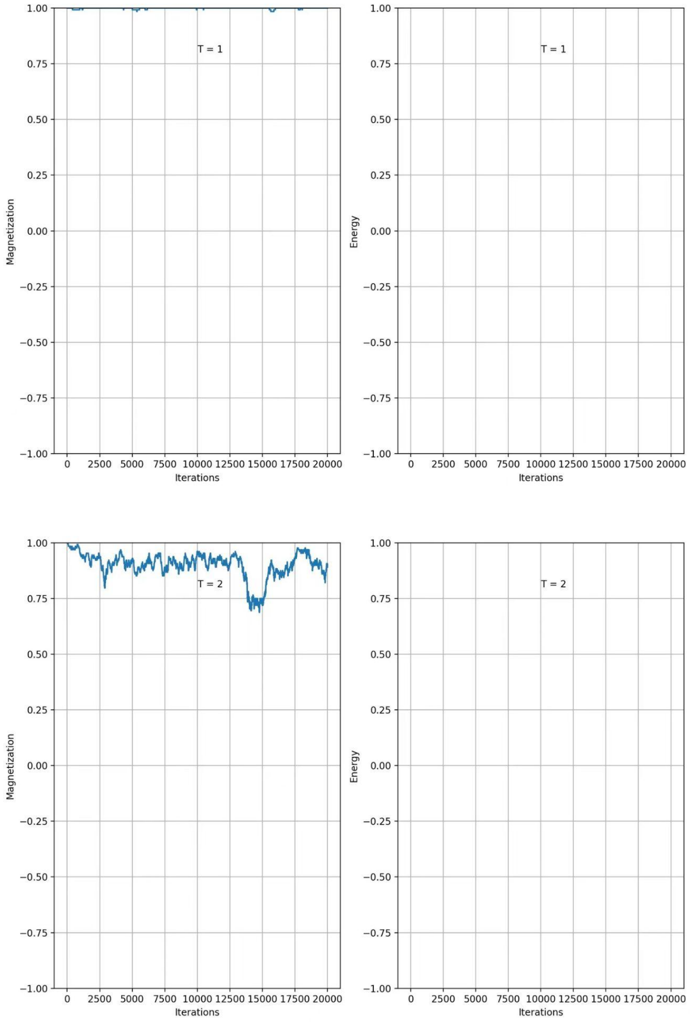
Variation with Temperature:



Berkeley
UNIVERSITY OF CALIFORNIA

Changes

Randomly generated >> homogeneous



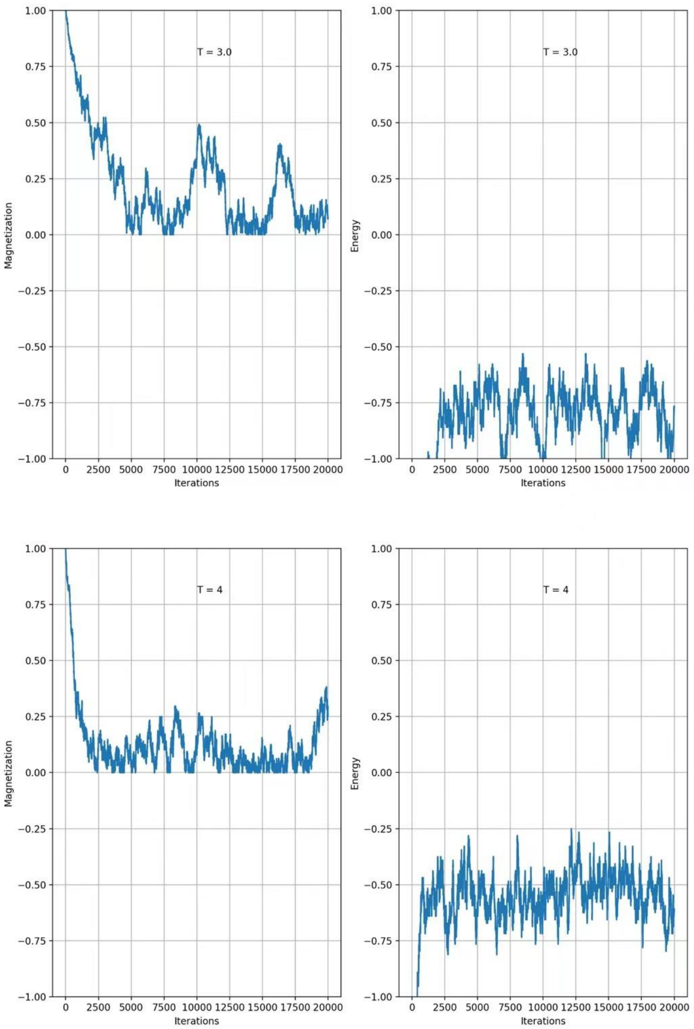
Berkeley
UNIVERSITY OF CALIFORNIA

Changes

Randomly generated >> homogeneous

Phase transition happens between T = 2 and T = 3.

$$\frac{kT_c}{J} = \frac{2}{\ln(1+\sqrt{2})} \approx 2.26918531421$$



Berkeley
UNIVERSITY OF CALIFORNIA

Evaluation

Limited grid size: 16 * 16 rather than 100 * 100

Changing initial conditions

[Critical temperature]

Berkeley
UNIVERSITY OF CALIFORNIA

Thank you!

Berkeley
UNIVERSITY OF CALIFORNIA