Finite size scaling of 2D Ising Model

Hasti Hojabr

Student ID: 97216040

Professor: Ali Sadeghi

Date: December 13, 2021

Contents

1 Introduction

1.1 The Ising Model

The Ising model is a model of a magnet. The idea behind it is that the magnetism of a bulk material is made up of combined magnetic dipole moments of many atomic spins within the material. The model postulate a lattice (usually a simple cubic lattice in three dimensions) with a magnetic dipole or spin on each site. The Hamiltonian of the system is given down below:

$$H = -J \sum_{<ij>} s_i s_j - B \sum_i s_i \tag{1}$$

s_i can take only two values ± 1 and J is exchange energy between the spins. B is an external magnetic field. $J > 0$ corresponds to ferromagnetic interactions (all the spins line up), while $J < 0$ corresponds to anti-ferromagnetic interactions (where neighbouring spins are oppositely aligned).
Interaction between atoms costs less energy for a site to be in the same state as its nearest neighbours. When temperature increases equilibrium is determined by minimizing Helmholtz function F so that the disordered configuration will be favoured at high temperature.

$$F = U - TS \tag{2}$$

Phase transition occurs at **Curie temperate** T_C.

$$T_C = 2.26918531421 \quad \frac{J}{k_b}$$

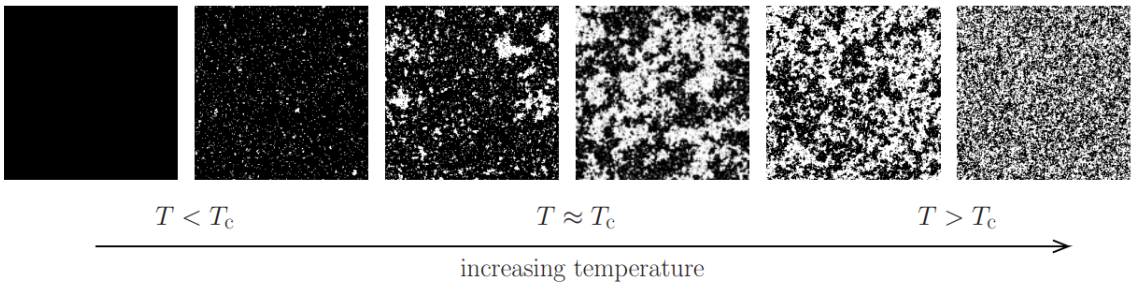

Figure 1: Two dimentional Ising Model lattice

The system spends most of its time in a subset of states with narrow range of energies and rarely makes transitions that change the energy of the system dramatically So we can consider only the states which differ from the present one by the flip of a single spin. So we have a single-spin-flip dynamics.

1.2 Phase Transition

Ising Model undergoes a phase transition. Magnetization in Ising Model is zero above the curie temperature and non-zero below it. Such parameters which are zero in discorded phase but non-zero in ordered phase, are called **Order Parameters**.
Phase transition in 2d Ising Model is a second order transition because we have second derivative of F in Susceptibility χ and so on.

Second order transitions are classified by their critical exponents, which characterize the behaviour at the critical point. Finding these exponents are crucial because of phenomenon called universality. Universality means that the values of critical exponents for a model (as well as a number of other measurable quantities) are independent of most of the parameters of our model such as J and B in Ising model. They depend on dimentinallity, periodic boundaries and number of dimensions of the order parameters. This phenomenon will make universality classes, in which we can study a simple model such as Ising model and our results will apply to more complicated models.

1.3 Observables

The partition function of the model is given down below. By calculating Z we can find average energy , heat capacity , average magnetisation and magnetic susceptibility.

$$Z = \sum_{s_i} e^{-\beta H} \tag{3}$$

$$< E >= -\frac{1}{Z}\frac{dZ}{d\beta} = -\frac{dlnZ}{d\beta} \tag{4}$$

$$C = \frac{d < E >}{dT} = k_B\beta^2(< E^2 > - < E >^2) \tag{5}$$

$$< M >=< \Sigma_i s_i > \tag{6}$$

$$\chi = \frac{\beta}{N}(< M^2 > - < M >^2) \tag{7}$$

2 Generating Data

2.1 Monte Carlo simulation

For Calculating Partition function we take the model and put it on a lattice of finite size, so that the partition function becomes a sum with finite number of terms.The method used for calculating the partition function of Ising model is Monte Carlo simulation with importance sampling. The idea behind it is to simulate the random thermal fluctuation of the system from state to state over the course of an experiment.

We need to find an appropriate random set of states according to the Boltzmann probability distribution.(Our model's probability distribution is Boltzmann probability distribution so we want to generate states which appear with probabilities given by the Boltzmann distribution.)

Markov processes is the generating engine for the set of states used. If we give Markov process a state μ it may or may not generate a new state ν. The probability of generating the new state is called the transition probability $P(\mu \rightarrow \nu)$. Markov process should follow these conditions:

1. Ergodicity : This condition is the requirement that it should be possible for Markov process to reach any state of he system from any other state , if we run it for long enough.
2. Be time independent.

3. Depend only on properties of the current states.

4. Detailed balance: the rate at which the system makes transitions into and out of any state μ must be equal. We must avoid limit cycles(rotating around some values.). Fulfilling Detailed balance also satisfies time-reversal symmetric of the model. So transition probability should follow this condition as well:

$$p_\mu P(\mu \to \nu) = p_\nu P(\nu \to \mu) \tag{8}$$

To find the right set of transition probabilities we also need to define two new features , selection probability $g(\mu \to \nu)$ and acceptance ratio A$(\mu \to \nu)$. Selection probability is probability that our algorithm will generate a new state and acceptance ratio says if we should accept the new state or not. So transition probability will be:

$$P(\mu \to \nu) = g(\mu \to \nu) A(\mu \to \nu) \tag{9}$$

2.2 The Metripolis Algorithm

One of the Monte Carlo methods is the Metripolis algorithm. The idea behind it is to use Boltzmann factors (The ratio of probabilities of two states) as a guide during the random generation of a subset of states to sample. It has the single-spin-flip dynamics as our system. With single-spin-flip dynamics there are N^2 different spins that we could flip so N^2 possible states ν to reach from μ.

$$g(\mu \to \nu) = \frac{1}{N^2} \tag{10}$$

SO selection probabilities will be:

$$\frac{P(\mu \to \nu)}{P(\nu \to \mu)} = \frac{g(\mu \to \nu) A(\mu \to \nu)}{g(\nu \to \mu) A(\nu \to \mu)} = \frac{A(\mu \to \nu)}{A(\nu \to \mu)} = e^{-\beta(E_\nu - E_\mu)} \tag{11}$$

Now we need to choose the acceptance ratios to satisfy equation 11. We can choose it like this:

$$A(\mu \to \nu) = A_0 e^{-\frac{1}{2}\beta \Delta E} \tag{12}$$

To satisfy Detailed balance, $A(\mu \to \nu)$ should be smaller than one. When $\Delta E < 0$ we have another Solution for $A(\mu \to \nu)$ which is equal one. So we will have:

$$A(\mu \to \nu) = \left\{ \begin{array}{ll} e^{-\beta \Delta E}, & 0 < \Delta E \\ 1, & \text{otherwise} \end{array} \right\} \tag{13}$$

We can also calculate $\Delta E = E_\nu - E_\mu$. Lets consider a spin α with its four neighbours λ. We can write its energy using formula 1.

$$E_\mu = -\alpha_\mu(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$$
$$E_\nu = -\alpha_\nu(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$$
$$\Delta E = (\alpha_\mu - \alpha_\nu)(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$$

if α_μ is +1 and we flip it then :

$$\Delta E = 2(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$$

if α_μ is -1 and we flip it then :

$$\Delta E = 2(\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4)$$

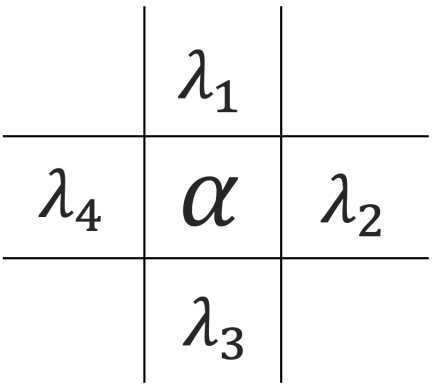

Figure 2: Neighbouring Sites of a given spin

So we will have:

$$\Delta E = 2\alpha_\mu \sum_{i=4} \lambda_i \tag{14}$$

Another important aspect of calculating ΔE is that all spins should have the same amount of neighbouring sites. For doing so we need to apply a periodic bounding condition. This way the spins on the edge of the lattice are neighbours to each other. For example the neighbours of site (i, j) will be neighbours to sites $(i, j+1), (i, j-1), (i+1, j)$ and $(i-1, j)$.

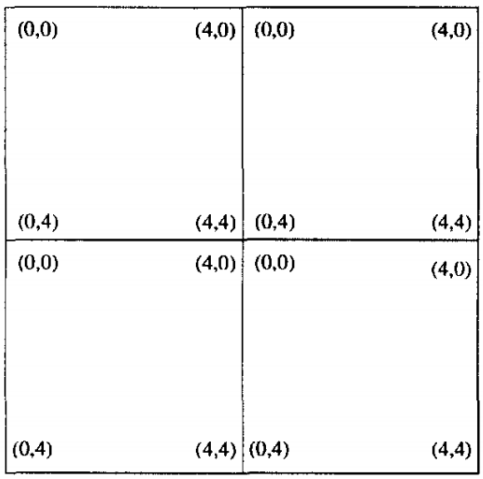

Figure 3: Periodic bounding condition

2.3 Results and Analysis

Start with a random state and choose a random spin and consider the possibility of flipping it. If $\Delta E \leq 0$, so the system's energy would decrease or remain unchanged, so flip the spin and generate the next state. If $\Delta E \geq 0$ decide at random whether to flip the spin or not with probability of the flip being $e^{-\beta \Delta E}$ .This quantity goes to 0 as the temperature T falls to 0, so that energetically unfavorable flips are unlikely in a cold lattice. The probability approaches 1 as T goes to infinity, which is why the model is such a seething mass of fluctuations at high temperature.
The new matrix represent the new state of the system. Then We can calculate average energy, heat

capacity ,average magnetisation and magnetic susceptibility and plot them.
First we have to run our simulation for a suitably long period of time until it has come to equilibrium at the temperature we are interested in—this period is called the equilibration time —and then we have to measure the quantity we are interested in over another suitably long period of time and average it, to evaluate the estimator of that quantity, we call this measurement. We must use the correct equilibration and measurment's steps, in order to do so we run our program for a large monte carlo step. When the fluctioatns are lowered we have reached the right equilibration step.

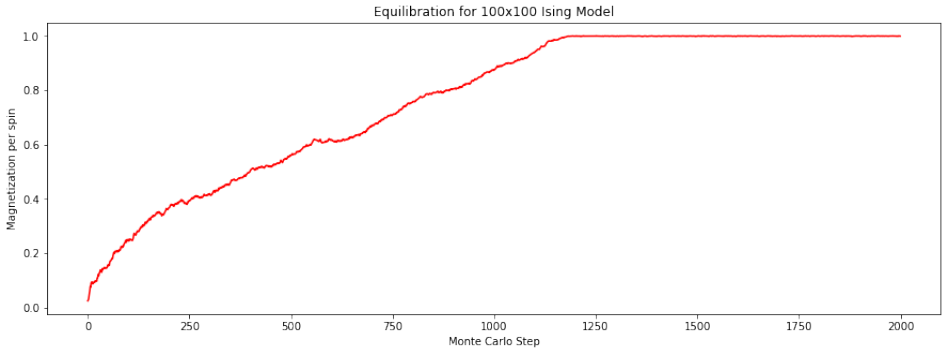

Figure 4: The magnetization per spin of 100x100 Ising model as a function of time (in Monte Carlo steps)

For $J = 1$ and $B = 0$ plots and figures are shown down below. We can clearly see the phase transition occurring around $2.269\frac{J}{k_b}$.

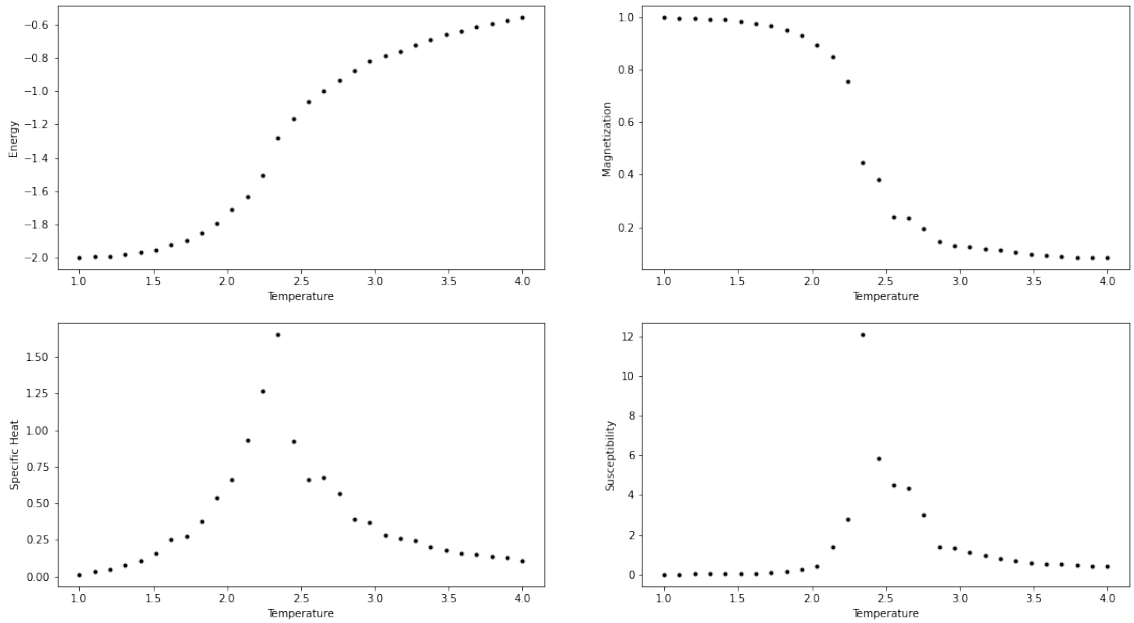

Figure 5: 20x20 Ising Model's Observables

3 Critical Temperature

Two-dimensional square lattice Ising model undergoes a phase transition at Curie temperature. Analytical solution given by Lars Onsager gives us the value :

$$T_c = 2.26918531421$$

3.1 Location of peaks

We can find T_c by finding the maximum location of susceptibility χ or specific heat C_v . This will give us an accuracy of 98.5% for 20 x 20 Ising in zero field for $(T - Cv)$ graph and an accuracy of 90% for $(T - \chi)$ graph. χ and Cv are sensitive to size of the lattice, this is shown in Fig 2, we have used two other methods to calculate T_c.

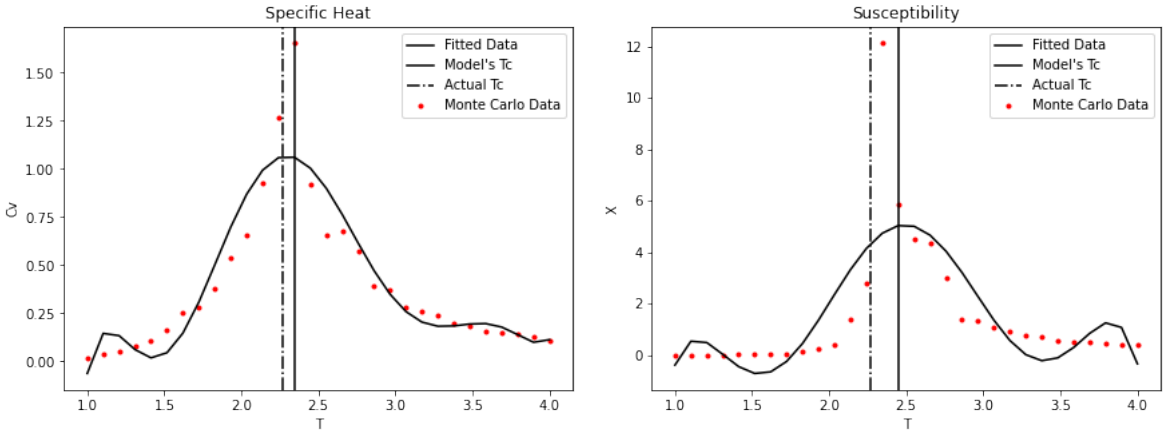

Figure 6: left: $(T - Cv)$ right: $(T - \chi)$

3.2 Binder Cumulant

Another frequently used method to determine the critical point is to use the intersection points of the Binder cumulants.

$$U_L = 1 - \frac{\langle m^4 \rangle_L}{\langle m^2 \rangle_L^2} \tag{15}$$

U_L varies weakly with temperature and linear dimension d near the crucial temperature and it stays close to a universal but nontrivial fixed-pint value U^*. So by using U_L for different pairs of L and by finding the interaction point of them we will have a good estimate of T_c.

U_L behaves as followed:

1. Symmetric Phase $(T > T_c)$: $U_L = 0$ as $L \to \infty$

2. Broken Phase $(T < T_c)$: $U_L = \frac{2}{3}$ as $L \to \infty$

3. Critical Point: $U_l \to U_L^*$ so that $0 < U_L^* < \frac{2}{3}$

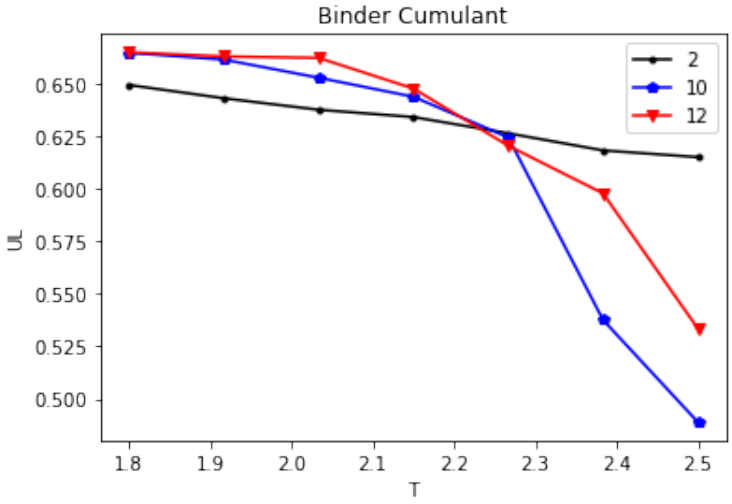

Figure 7: Binder Cumulant for different lattice sizes

So using three pairs of L and equation

$$U_L = U_{L'}$$

We will find $Tc = 2.26666667$ with an accuracy of 99.89%. Fig 8.

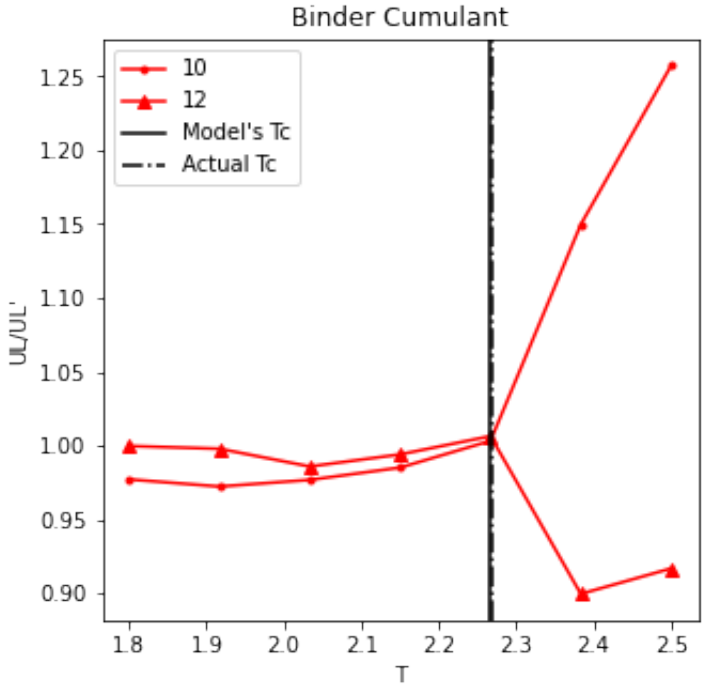

Figure 8: Binder Cumulant

3.3 The renormalization group

Renormalization group (RG) refers to a formal apparatus that allows systematic investigation of the changes of a physical system as viewed at different scales. In RG we use a method called blocking. Given it an initial condition for example Ising 32x32, it will give us a coarse grained lattice. Two usual methods are 'majority rule' and 'decimation procedure' . The size of the lattice decreases by recalling factor called b. The number of spins in the system decrease on blocking by b^d which d is dimensionality of the system. If we use $b = 2$ in our previous example and decimation procedure, in which we only consider top left spin, we will get a coarse grained lattice.
Correlation lenght of the blocked system is:

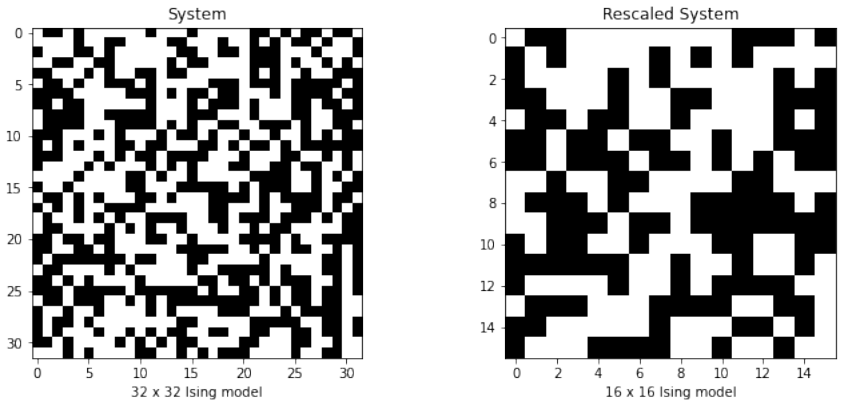

Figure 9: Decimation procedure on 32x32 two dimensional Ising Model.

$$\xi^{'} = \xi / b$$

Its like the blocked states are of an Ising model at a temperature $T^{'}$. At critical temperature correlation lenght becomes infinite so two systems have the same temperature $T = T^{'} = T_c$. So they have the same internal energy per spin.
We can find critical temperature by performing a Monte Carlo simulation at temperature the internal energy E. We used decimation procedure to block some spins. Then we calculate internal energy u for blocked system. Then we Plot (E-u) and find critical temperature by locating the crossing point.

4 Critical Exponents

We have some critical exponents that can be derived our of Monte Carlo simulation and some are related to each other through scaling laws.
We define a new parameter called the reduced temperature t:

$$t = T - T_c$$

One way of finding critical exponents is Finite size scaling. Another is renormalization group method given by Swendsen in 1979. Renormalization group is faster and more accurate than finite size scaling But we can measure the errors. In finite size scaling we can indeed measure errors but the computational time is long due to the need of different measurements of Ising model's quantities in different lattice sizes.

4.1 Finite Size Scaling

The finite size scaling method is a way of extracting values for critical exponents by observing how measured quantities vary as the size L of the system studied changes. We can see effect of L on magnetization and susceptibility in Fig 10. With right critical exponents, magnetization for different L should collapse into the same curve.

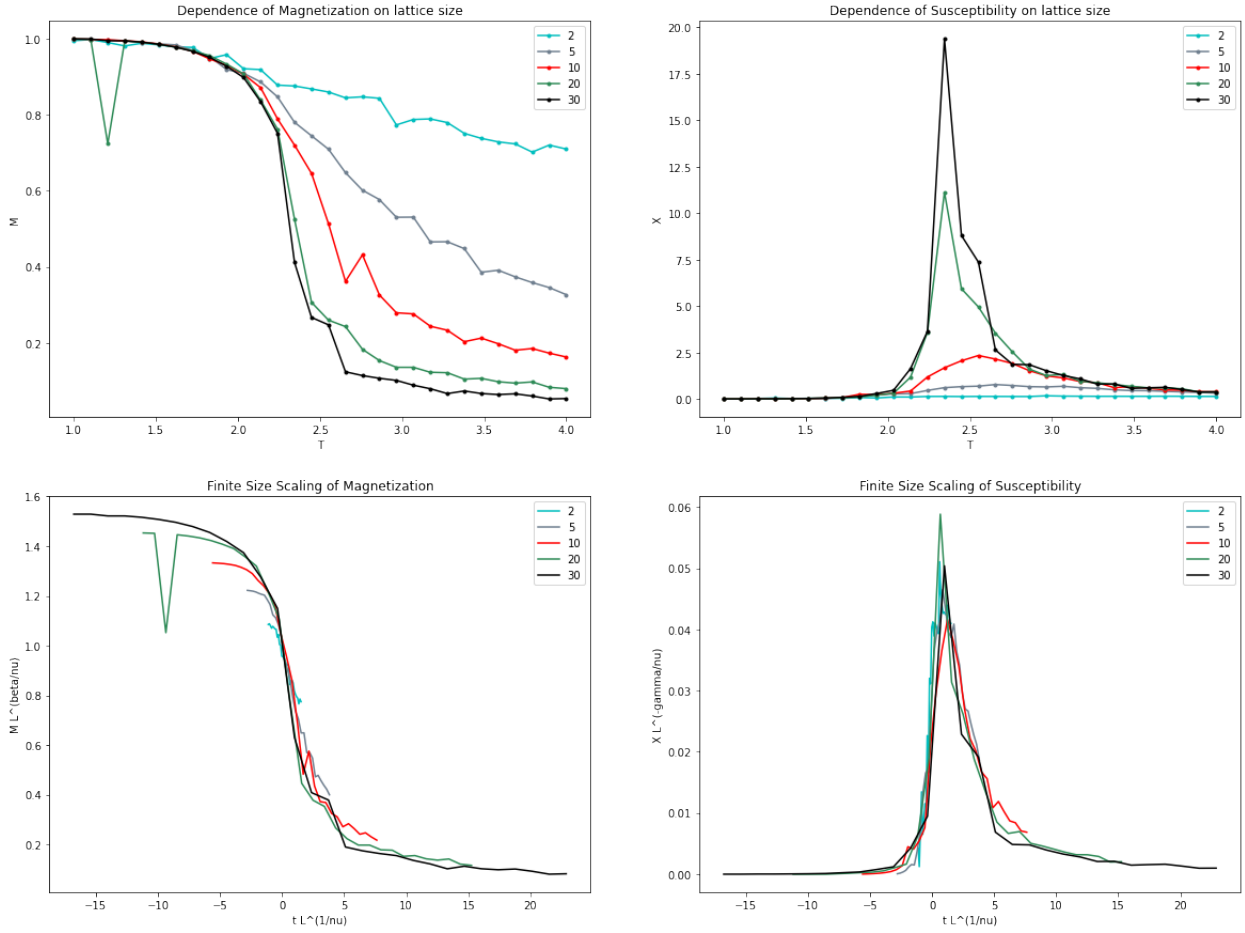

Figure 10: Finite Size effects

4.1.1 Nu ν

Nu is Correlation length's exponent.

$$\xi = |T - T_c|^{-\nu}$$

Near critical temperature we know that correlation length will be on the order of lattice size L.

$$\xi \sim L$$

So by finding T near T_c for different lattice sizes (L=10, 15, 20, 30 & 50) and plot equation 1. We will get ν.

$$|T - T_c|^{-\nu} = L \tag{16}$$

The result from fit is $\nu = 0.91$. The analytical value of ν is 1. So the accuracy is about 91.2 percent.

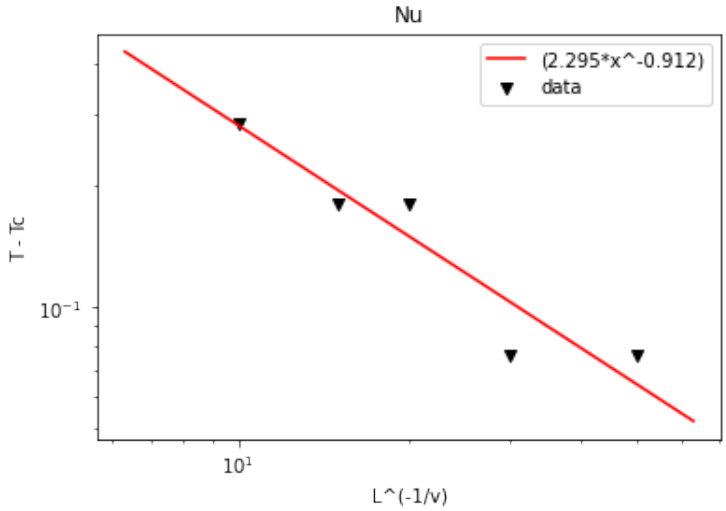

Figure 11: $(L^{\frac{-1}{\nu}} - t)$

4.1.2 Beta β

In a Ising model for T<Tc, Probability distribution P(M) of a finite lattice has two peaks. A spontaneous magnetization $\pm|M_s p|$ appears. P(M) has $+|M_s p|$ peak where the lattice has uniformly negative magnetization and it has $-|M_s p|$ for when the lattice has positive magnetization. In between theses two states P(M) has a minimum value.(Fig 12) The system spontaneously makes excursions from one state to another So we will use root mean square order parameter to avoid these fluctuations. On a finite and infinite lattice M_{rms} is non-zero in every temperature. We can make an approximation for M_{rms} as followed:

$$M_{rms} = \sqrt{\langle M_2 \rangle_T} = \frac{1}{N} \left(\sum_{i,j=1}^{N} \langle S_i S_j \rangle \right)_T^{\frac{1}{2}} \tag{17}$$

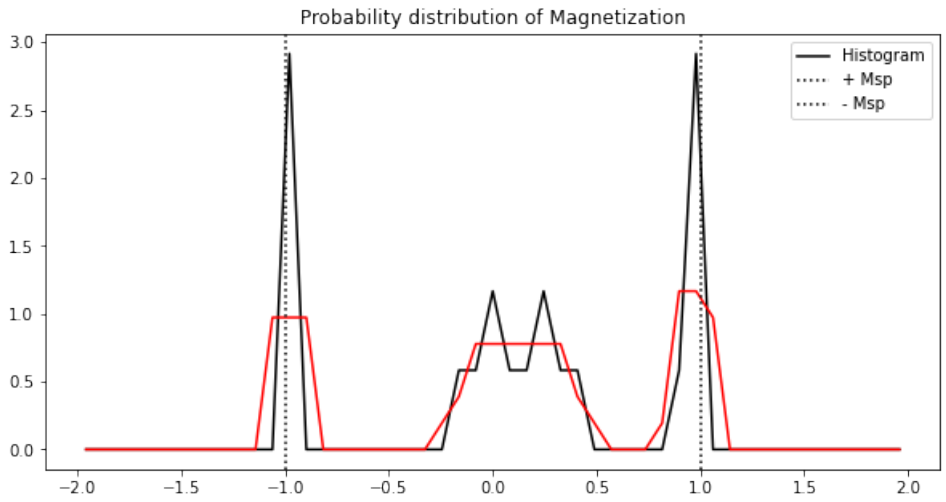

Figure 12: Probibility distribution P(M) for 20x20 Ising Model ($T < T_c$)

In an infinite system at critical temperature, correlation function G can be approximated.

$$G(\mathbf{R}_{ij}) \xrightarrow{|\mathbf{R}_{ij}|\to\infty} G|\mathbf{R}_{ij}|^{-(d-2+\eta)} \tag{18}$$

Writing G for finite lattice of two dimensional Ising Model.

$$\sum_{i,j=1}^{N} \langle S_i S_j \rangle \propto \int_0^{L/2} rG(r)dr \propto \int_0^{L/2} rG(r)dr$$

Submitting Equation 18:

$$\int_0^{L/2} G|\mathbf{r}_{ij}|^{-(2-2+\eta)} r_{ij} dr_{ij} = r_{ij}^{2-\eta}\,|_0^{L/2} \propto L^{2-\eta}$$

$$M_{rms} \propto L^{2-\eta} \tag{19}$$

Using scaling laws:

$$2-\eta = \frac{\gamma}{\nu} \tag{20}$$

$$d\nu = 2\beta + \gamma \tag{21}$$

We will have:

$$M_{rms} \propto L^{\frac{\frac{\gamma}{\nu}-d}{2}}$$

$$M_{rms}|_{T=T_c} \propto L^{-\frac{-\beta}{\nu}} \tag{22}$$

Running Ising model for L=2,5,10,15 & 20 and finding value of M in critical temperature, we can get value of M_{rms} using Eq. 22. Using results of our previous calculation for ν, we plot Eq.22 and find the slope.

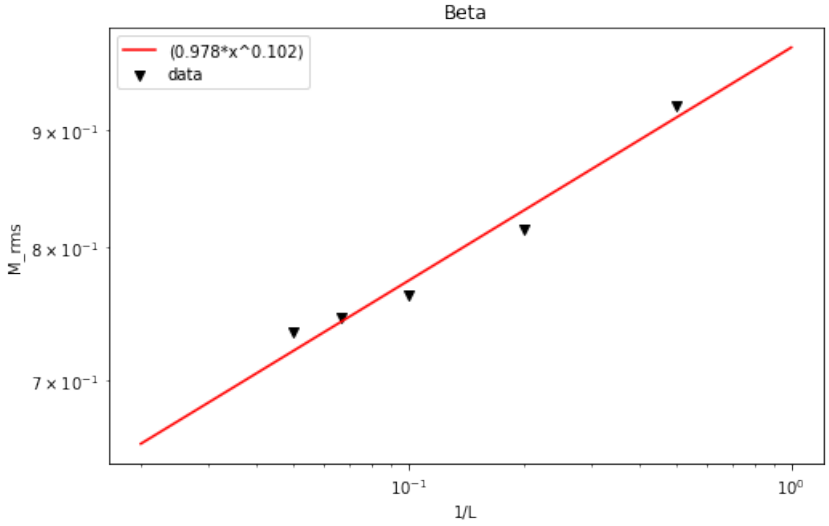

Figure 13: Magnetization's critical exponent ($L - M_{rms}$)

The result from fit is $\beta = 0.1$. The analytical value of β is $\frac{1}{8}$. So the accuracy is about 83 percent.

4.1.3 Alpha α

The scaling relation for specific heat is:

$$c \propto |T - T_c|^{-\alpha} \tag{23}$$

We know near critical temperature $\xi \propto L$ and using equation 1 from 4.1.1 we get:

$$L^{1/\nu} = |T - T_c| \rightarrow$$

$$C_v|_{T=T_C} = L^{\frac{\alpha}{\nu}} \tag{24}$$

By finding c for different L and then locating max specific heat c_{max} which happens at critical temperature, we can plot Eq. 24 and find α.

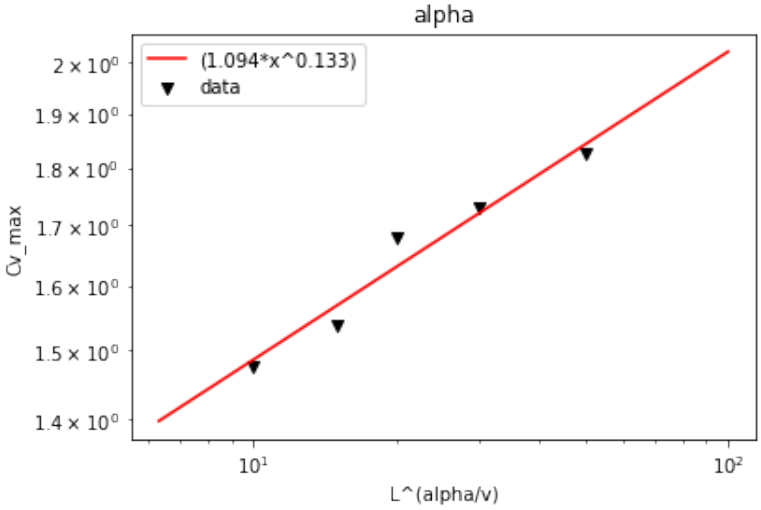

Figure 14: Specific Heat's critical exponent ($L^{\frac{\alpha}{\nu}} - Cv$)

The result from fit is $\alpha = 0.13$. The analytical value of α is zero. So the accuracy is about 87 percent.

4.1.4 Gamma γ

The scaling relation for susceptibility is:

$$\chi \propto |T - T_c|^{-\gamma} \tag{25}$$

Near critical temperature:

$$\chi|_{T=T_C} = L^{\frac{\gamma}{\nu}} \tag{26}$$

By finding χ for different L and then locating the peak χ_{max} which happens at critical temperature, we can plot Eq. 26 and find γ.

The result from fit is $\gamma = 1.7$. The analytical value of γ is $\frac{7}{4}$. So the accuracy is about 96.34 percent.

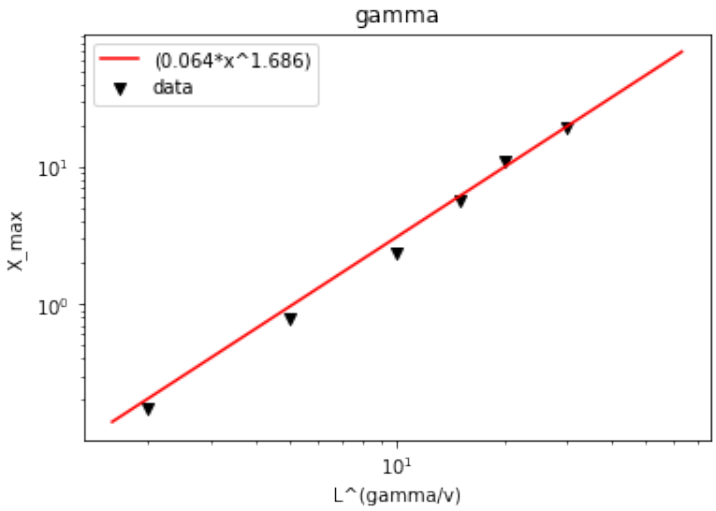

Figure 15: Susceptibility's critical exponent($L^{\frac{\gamma}{\nu}} - X$)

5 Appendix

5.1 2d Ising model simulation

```
def Ising(N,Ti,Tf,n,equlibration_step,measurment_step):

    #Lattice Size , initial Temperature , final Temperature , Temperature steps , Equlibration Step , Measurment Step

    def Random_Matrix(N):
        return np.random.choice([1,-1], size=(N,N))

    def Metropolis(State,t,N,step):
        m,mm,ee,e=[],[],[],[]
        Eng,EngEng,Mag,MagMag = [],[],[],[]
        for s in range(step):
            En=0
            for i in range(N*N):
                x, y = np.random.randint(0,N) , np.random.randint(0,N)
                Neighbour = State[(x-1)%N,y]+State[x,(y-1)%N]+State[(x+1)%N,y]+State[x,(y+1)%N]
                dE = 2*State[x,y]*Neighbour
                if dE<0:
                    State[x,y] =- State[x,y]
                elif np.random.random(1)[0] < np.exp(-(beta[t])*dE):
                    State[x,y] =- State[x,y]
                En -= State[x,y]*Neighbour
            M = np.sum(State)
            m.append(abs(M))
            mm.append(abs(M)**2)
            e.append(En/2)
            ee.append(En**2/4)

        Mag,MagMag,Eng,EngEng = np.mean(m),np.mean(mm),np.mean(e),np.mean(ee)

        C=(beta[t]**2)*(EngEng-Eng**2)/(N**2)
        x = (beta[t])*(MagMag-Mag**2)/(N**2)
        return State , Eng/(N**2) , C , sum(m)/step/(N**2) , x

    E,X,Cv,M,U = np.zeros(n),np.zeros(n),np.zeros(n),np.zeros(n),np.zeros(n)
    T = np.linspace(Ti,Tf,n)
    beta = 1/T

    for t in range(0,n):
        State = Random_Matrix(N)
        Metropolis(State,t,N,equlibration_step) #equlibration
        State,e,cv,m,x = Metropolis(State,t,N,measurment_step)    #measurment

```

```
        E[t],M[t],Cv[t],X[t] = e,m,cv,x

    fig = plt.figure(figsize=(18, 10))
    name = ['Energy','Magnetization','Specific Heat','Susceptibility']

    for i,ax,name in zip([E,abs(M),Cv,X],range(1,5),name):
        ax =  fig.add_subplot(2, 2, ax )
        ax.scatter(T,i,color = 'k',marker = '.')
        ax.set_xlabel('Temperature')
        ax.set_ylabel(name)

    return plt.show(),E,Cv,M,X,T
```

5.2 Binder Cumulant

```
def Binder(N,Ti,Tf,n,equlibration_step,measurment_step):

    #Lattice Size , initial Temperature , final Temperature , Temperature
     ↪ steps , Equlibration Step , Measurment Step

    def Random_Matrix(N):
        return np.random.choice([1,-1], size=(N,N))

    def Metropolis(State,t,N,step):
        mm,mmmm=[],[]
        MagMag,MagMagMagMag = [],[]

        for s in range(step):
            En=0
            for i in range(N*N):
                x, y = np.random.randint(0,N) , np.random.randint(0,N)
                Neighbour =
                 ↪  State[(x-1)%N,y]+State[x,(y-1)%N]+State[(x+1)%N,y]+State[x,(y+1)%N]
                dE = 2*State[x,y]*Neighbour
                if dE<0:
                    State[x,y] =- State[x,y]
                elif np.random.random(1)[0] < np.exp(-(beta[t])*dE):
                    State[x,y] =- State[x,y]
            M = np.sum(State)

            mm.append(abs(M)**2)
            mmmm.append(abs(M)**4)

        MagMag, MagMagMagMag  = np.mean(mm), np.mean(mmmm)
        U=(1-(MagMagMagMag)/(3*MagMag**2))

        return State  , U

    U = np.zeros(n)
```

```
    T = np.linspace(Ti,Tf,n)
    beta = 1/T

    for t in range(0,n):
        State = Random_Matrix(N)
        Metropolis(State,t,N,equlibration_step)            #equlibration
        State,u = Metropolis(State,t,N,measurment_step)    #measurment
        U[t] = u
    fig = plt.figure(figsize=(5, 5))
    plt.scatter(T,U,color = 'k',marker = '.')
    return plt.show(),U,T
```

```
plt.plot(T,u2,label='2',marker='.',color='k')
plt.plot(T,u10,label='10',marker='p',color='b')
plt.plot(T,u12,label='12',marker='v',color='r')
plt.title('Binder Cumulant')
plt.xlabel('T')
plt.ylabel("UL")
plt.legend()
plt.show()
```

```
fig = plt.figure(figsize=(5, 5))

binder1,binder2=u2/u10,u10/u12
plt.plot(T,u2/u10,label='2/10',marker='.',color='r')
plt.plot(T,u10/u12,label='10/12',marker='^',color='r')

idx=np.argwhere(np.diff(np.sign(binder1 - binder2 )) != 0).reshape(-1)
Tc = T[idx]
loss = (abs(Tc-2.26918531421)/2.26918531421)*100
plt.axvline(x=Tc, color='k', linestyle='-',label="Model's Tc")
plt.axvline(x=2.26918531421, color='k', linestyle='-.',label="Actual Tc")
plt.title('Binder Cumulant')
plt.xlabel('T')
plt.ylabel("UL/UL'")
plt.legend(loc='upper left')
plt.show()
print('Critical Tempreture is : {} with accuracy of :{}'.
 ↪ format(Tc,100-loss))
```

5.3 Critical Exponents

5.3.1 Collapse

```
fig = plt.figure(figsize=(20, 15))
M,C,X,L=[M2,M5,M10,M20,M30_fixed],[Cv2,Cv5,Cv10,Cv20_fixed,Cv30],[X2,X5,X10,X20_fixed,X

plt.subplot(2, 2, 1)
for Q,n,c in
 ↪ zip(M,['2','5','10','20','30'],['c','slategray','r','seagreen','k']):
```

```
    plt.plot(T,Q,'.-',label=n,color=c)
plt.title('Dependence of Magnetization on lattice size')
plt.xlabel('T')
plt.ylabel('M')
plt.legend()

plt.subplot(2, 2, 2)
for Q,n,c in
 ↪ zip(X,['2','5','10','20','30'],['c','slategray','r','seagreen','k']):
    plt.plot(T,Q,'.-',label=n,color=c)
plt.title('Dependence of Susceptibility on lattice size')
plt.xlabel('T')
plt.ylabel('X')
plt.legend()

Tc,nu,beta,alpha,gamma=2.26918531421,1,1/8,0,7/4

plt.subplot(2, 2, 3)
for Q,l,n,c in
 ↪ zip(M,L,['2','5','10','20','30'],['c','slategray','r','seagreen','k']):
    t=(T-Tc)/Tc*l**(1/nu)
    m_fsc=Q*l**(beta/nu)
    plt.plot(t,m_fsc,label=n,color=c)
    plt.title('Finite Size Scaling of Magnetization')
    plt.xlabel('t L^(1/nu)')
    plt.ylabel('M L^(beta/nu)')
    plt.legend()


plt.subplot(2, 2, 4)
for Q,l,n,c in
 ↪ zip(X,L,['2','5','10','20','30'],['c','slategray','r','seagreen','k']):
    t=(T-Tc)/Tc*l**(1/nu)
    x_fsc=Q*l**(-gamma/nu)
    plt.plot(t,x_fsc,label=n,color=c)
    plt.title('Finite Size Scaling of Susceptibility')
    plt.xlabel('t L^(1/nu)')
    plt.ylabel('X L^(-gamma/nu)')
plt.legend()
plt.show()
```

5.3.2 Exponents

Beta

```
ML=[]
for i in [M2,M5,M10,M15,M20]:
    ML.append((i[0]+i[29])/2)
L=[2,5,10,15,20]
Mrms = [(lambda x: x**(0.5))(x) for x in ML]
```

```
rL=1/np.array(L)
fig = plt.figure(figsize=(8,5))
ax=plt.gca()
ax.scatter(rL,Mrms,c="k",marker='v', label='data')
ax.set_yscale('log')
ax.set_xscale('log')
newX = np.logspace(-1.7, -0, base=10)
def myExpFunc(x, a, b):
    return a * np.power(x, b)
popt, pcov = curve_fit(myExpFunc, rL, Mrms)
plt.plot(newX, myExpFunc(newX, *popt), 'r-',
 ↪  label="({0:.3f}*x^{1:.3f})".format(*popt))
print(1/8)
print('-Beta/v = {}'.format(popt[1]))
loss = (abs(popt[1]-(1/8))/(1/8))*100
plt.title('Beta')
plt.ylabel('M_rms')
plt.xlabel('1/L')
print('Beta is : {} with accuracy of :{}'.
 ↪  format(round(abs(popt[1]),2),100-loss))
plt.legend()
plt.show()
```

Gamma

```
C=[Cv5,Cv10,Cv15,Cv20_fixed,Cv30,Cv50]
cmax=[]
for i in C:
    cmax.append(max(i))
L=[5,10,15,20,30,50]
from scipy.optimize import curve_fit
ax=plt.gca()
ax.scatter(L[1:],cmax[1:],c="k",marker='v', label='data')
ax.set_yscale('log')
ax.set_xscale('log')
newX = np.logspace(0.8, 2., base=10)
def function(x,a,b,v):
    return a * np.power(x, b)
popt, pcov = curve_fit(function, L[1:],cmax[1:])
plt.plot(newX, function(newX, *popt), 'r-',
 ↪  label="({0:.3f}*x^{1:.3f})".format(*popt))
print('alpha/v = {}'.format(popt[1]))
loss = (abs(popt[1]))*100
print('alpha is : {} with accuracy of :{}'.
 ↪  format(round(abs(popt[1]),2),100-loss))
plt.title('alpha')
plt.xlabel('L^(alpha/v)')
plt.ylabel('Cv_max')
plt.legend()
plt.show()
```

Nu

```
X=[X10,X15,X20_fixed,X30_fixed,X50]
L=[10,15,20,30,50]
xmax=[]
for i in X:
    xmax.append(max(i))
t=[]
for x in X:
    _,_,data,i=Poly_fit(T,x,14)
    t.append(T[i])
t_c=np.array(t)-2.269
ax=plt.gca()
ax.scatter(L,t_c,c="k",marker='v', label='data')
ax.set_yscale('log')
ax.set_xscale('log')
newX = np.logspace(0.8, 1.8, base=10)
def function(x,a,b,v):
    return a * np.power(x, b)
print(1)
popt, pcov = curve_fit(function, L,t_c)
plt.plot(newX, function(newX, *popt), 'r-',
 ↪ label="({0:.3f}*x^{1:.3f})".format(*popt))
print('-1/v = {}'.format(popt[1]))
loss = (abs(popt[1]+1))*100
print('nu is : {} with accuracy of :{}'.
 ↪ format(round(abs(popt[1]),2),100-loss))
plt.title('Nu')
plt.xlabel('L^(-1/v)')
plt.ylabel('T - Tc')
plt.legend()
plt.show()
```

6 References

Newman, Mark E. J.; Barkema, Gerard T., Monte Carlo methods in statistical physics, (1999)
Rubinstein Reuven Y. ; Kroese Dirk P. , Simulation and the Monte Carlo method , Third Edition
Schroeder Daniel V. , An Introduction to Thermal Physics
Blundell, Stephen J.; Blundell , Katherine M. , Concepts in Thermal Physics , Second edition
Binder, K ; Heermann, D.W, Monte Carlo Simulation in Statistical Physics AN Introduction, 4th Ed.
K. Binder: X.Phys. B 43,119 (1981)
Rubinstein Reuven Y. ; Kroese Dirk P. , Simulation and the Monte Carlo method , Third Edition
Schroeder Daniel V. , An Introduction to Thermal Physics

R.H. Swendsen , Monte Carlo Renormalization Group, 10.1103