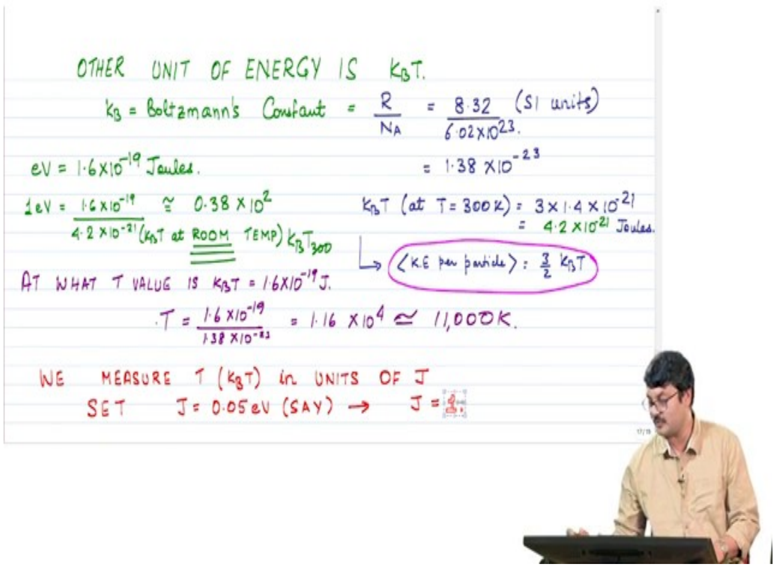
Computational Physics

Dr. Apratim Chatterji
Dr. Prasenjit Ghosh
Department of Physics
Indian Institute of Science Education and Research, Pune

Lecture - 19
Monte Carlo Simulation Algorithm & Implementation
Part 02

Now, just to remind you that electron volt is 1.6 into 12 10 to the power minus 19 Joules, right.

(Refer Slide Time: 00:16)



OTHER UNIT OF ENERGY IS K_BT.

k_B = Boltzmann's Constant $= \frac{R}{N_A} = \frac{8.32}{6.02\times10^{23}}$ (SI units)

$= 1.38\times10^{-23}$

eV $= 1.6\times10^{-19}$ Joules.

$1\,\text{eV} = \frac{1.6\times10^{-19}}{4.2\times10^{-21}}$ (K_BT at $\underline{\underline{\text{ROOM}}}$ TEMP) $\simeq 0.38\times10^{2}\ K_BT_{300}$

K_BT (at $T = 300\,K$) $= 3\times1.4\times10^{-21}$
$= 4.2\times10^{-21}$ Joules.

$\hookrightarrow \boxed{\langle \text{K.E per particle} \rangle : \frac{3}{2} K_BT}$

AT WHAT T VALUE IS $K_BT = 1.6\times10^{-19}$ J.

$T = \frac{1.6\times10^{-19}}{1.38\times10^{-23}} = 1.16\times10^{4} \simeq 11{,}000$ K.

WE MEASURE T (K_BT) in UNITS OF J

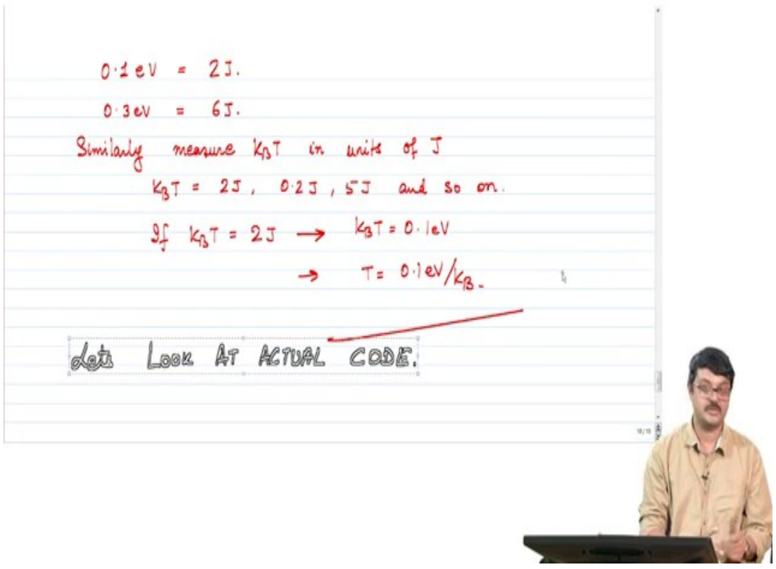
SET J = 0.05 eV (SAY) $\rightarrow$ J = [illegible]

Now, if you want to express 1 electron volt in terms of K B T, K B T of room temperature, right then you would say that, so 1 electron volt is 1.6 into 10 to the power minus 19 into 4.2 into 10, 10 to the power minus 21. Do the calculation, and what you get is 0.38 into 10 to the power 22 K B T 300, right. If you change temperature the value of K B T is going to change, remember that.

So, a question would be. So, we were discussing units and at what value of temperature is K B T equal to 1.6 into 10 to power minus 19 Joules, 1 electron volt and if you calculate that that is essentially this calculation 1.6 into 10 to the power minus 19 by K B. And what you get out of it is approximately 11,000 Kelvin it is a really high temperature all everything would melt and vaporize at that temperature, .

So, the point I am trying to make is we cannot be working with different units. What we are going to do is measure K B T or the thermal energy the temperature you divide K B T by K B you get the temperature. We shall measure K B T in units of J, J be the coupling constant minus J S i dot S j. What does it mean by that? Now, suppose it just suppose that for a particular material which can be modelled by the Ising model say, J is equal to minus 0.05 electron volt say and we say J equal to 1. So, we measure energy in units of J. So, we were measuring it in terms of electron volts, we say that we are measuring we are going to measure energy in units of J, where J has the value 0.05 electron volt.

(Refer Slide Time: 02:54)



$0.1\,eV = 2J.$

$0.3\,eV = 6J.$

Similarly measure K_BT in units of J

$K_BT = 2J,\ 0.2J,\ 5J$ and so on.

If $K_BT = 2J \rightarrow K_BT = 0.1\,eV$

$\rightarrow T = 0.1\,eV/K_B.$

Lets Look At Actual Code.

Then what would we get? 0.1 electron volt we will call it 2J, J has been set to 1, but I mean suppose J equal to 0.05 electron volt, in 0.1 electron volt equal to 2J. 0.3 electron volts equal to 6J. Similarly, we can measure K B T. So, here what were we doing? We were measuring some unit of energy electron volt in terms of J. We can measure K B T in terms of J. So, we can say that K B T equal to 2 J say or K B T equal to 0.2 J or 5 J and so on, right.

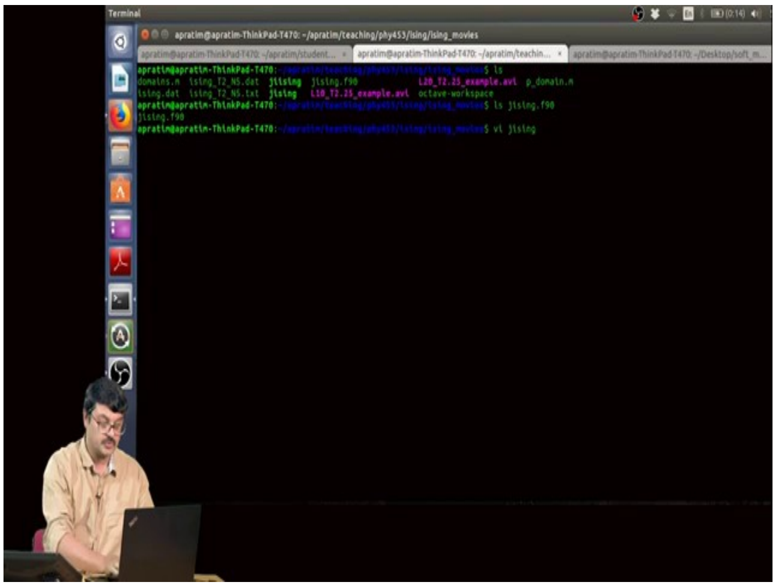
So, what does it mean to say that K B T is 2 J? If K B T is 2 J, it means K B T equal to 0.1 electron volts and from there you can basically get the value of T to be 0.01 electron volt by K B which will give you the units of temperature, ok. With this background where we have set all the units all the basically Bohr Magneton h cross etcetera are sitting inside the J, we are basically going to measure magnetization in terms of spin plus 1 and minus 1. And we are going to measure the temperature, the thermal energy. So, I am using this equivalently, but K

B T is thermal energy T has units of temperature, but you take thermal energy divided by K B you get T.

We shall be measuring temperature in units of not electron volt, but in units of J, where we shall set J equal to 1 and then figure out at what temperature, at what value of temperature in units of J does the system an Ising model get demagnetized or loses its ferromagnetic behaviour; at what value of temperature do we get that. So, that is basically where we are heading, ok.

With this background let us start looking at the actual computer code. All this question that we have had how do we actually implement it? And then, when you see results out from the code we are going to discuss this, look at the effect of thermal fluctuations thermal energy you will have a better understanding of what I told on this screen, right. So, let us actually head to the actual computer.

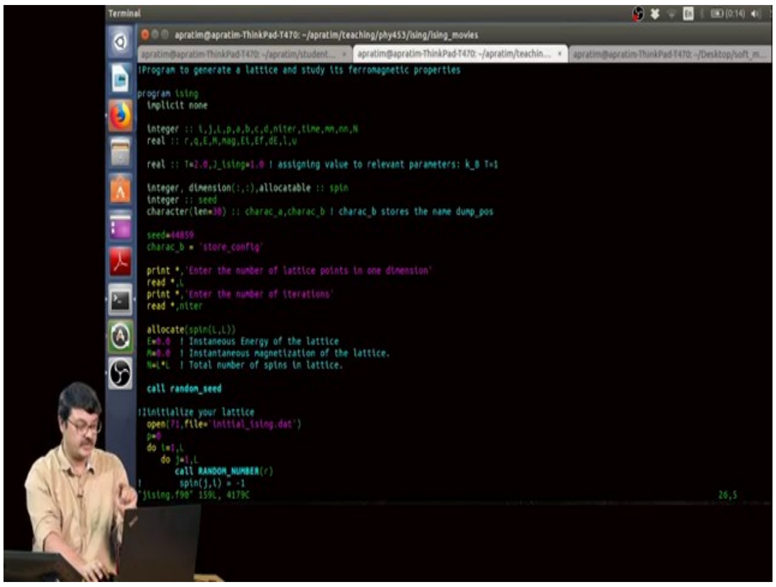
(Refer Slide Time: 05:32)


Terminal
apratim@apratim-ThinkPad-T470: ~/apratim/teaching/phy453/ising/ising_movies
apratim@apratim-ThinkPad-T470: ~/apratim/student...
apratim@apratim-ThinkPad-T470: ~/apratim/teachin...
apratim@apratim-ThinkPad-T470: ~/Desktop/soft_m...

```
apratim@apratim-ThinkPad-T470:~/apratim/teaching/phy453/ising/ising_movies$ ls
domains.m  ising_T2_N5.dat  jiising  jising.f90            L20_T2.25_example.avi  p_domain.m
ising.dat  ising_T2_N5.txt  jising   L10_T2.25_example.avi  octave-workspace
apratim@apratim-ThinkPad-T470:~/apratim/teaching/phy453/ising/ising_movies$ ls jising.f90
jising.f90
apratim@apratim-ThinkPad-T470:~/apratim/teaching/phy453/ising/ising_movies$ vi jising
```

Now, basically the name of my code is jising dot f90, ok. So, I am in the appropriate directory. And I am going to like to use the vi editor, you can use emacs or whatever is a editor of your choice and I shall show you the code; jising dot f90.

(Refer Slide Time: 05:59)



```
!Program to generate a lattice and study its ferromagnetic properties

program ising
  implicit none

  integer :: i,j,L,p,a,b,c,d,niter,time,mm,nn,N
  real :: r,q,E,M,mag,Ei,Ef,dE,l,u

  real :: T=2.0,J_ising=1.0 ! assigning value to relevant parameters: k_B T=1

  integer, dimension(:,:),allocatable :: spin
  integer :: seed
  character(len=30) :: charac_a,charac_b ! charac_b stores the name dump_pos

  seed=44859
  charac_b = 'store_config'

  print *,'Enter the number of lattice points in one dimension'
  read *,L
  print *,'Enter the number of iterations'
  read *,niter

  allocate(spin(L,L))
  E=0.0  ! Instaneous Energy of the lattice
  M=0.0  ! Instantaneous magnetization of the lattice.
  N=L*L  ! Total number of spins in lattice.

  call random_seed

!Initialize your lattice
  open(71,file='initial_ising.dat')
  p=0
  do i=1,L
     do j=1,L
        call RANDOM_NUMBER(r)
!        spin(j,i) = -1
"jising.f90" 159L, 4179C                                             20,5
```

It is a program it is a Fortran 90 code, name is Ising, a program Ising I have defined implicit none which is a safe thing to do. Of I am. So, basically you do not end up doing any miscalculations all the variables you have to define by hand and to do your calculations. Here I have defined some variables i, j, L and so on so forth, which I shall use basically to count dummy variables as we go proceed choose different spins of the lattice or to monitor, the number of iterations and so on so forth.

What is important is you can give them i, j k L whatever you want integers typically. Here L is essentially L is the size of the lattice which one shall read here, ok. So, you can read L and change the size of the lattice which we shall be using. The other important quantity is neater, is basically the number of iterations that it will be running at a particular temperature. So, at the moment we shall need to understand how these thermodynamic quantities fluctuate due to at a particular temperature.

Later we will be calculating the phase diagram, we will keep things simple and calculate things one by one, increase the complexity one by one that is what I mean by that. And the quantities of interest are T equal to 2. So, I have set temperature T equal 2, where jising which is the same as J when we were discussing it on board is essentially I was well set to one I can change temperature to 2.1, 2, 2.5, 0.3. So, you are basically changing temperature in units of J as I discussed previously, ok.

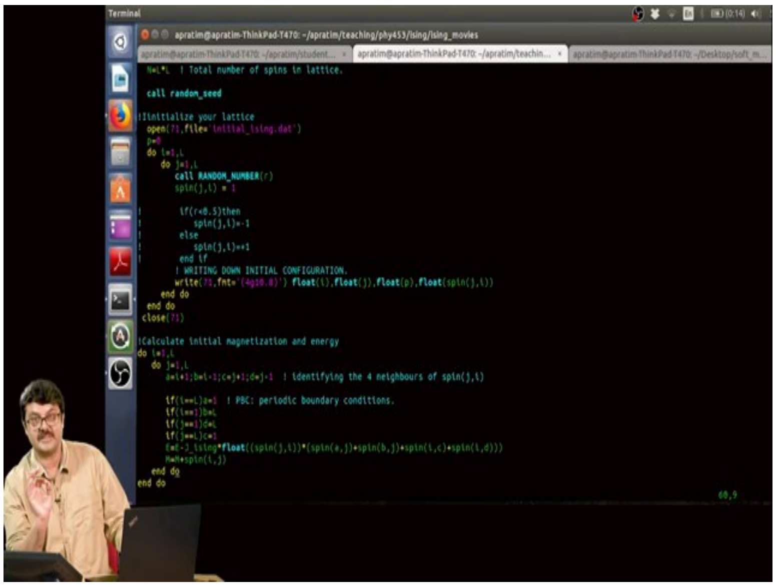
So, you have to give a random number seed here, and the spins plus 1 and minus 1 and basically defining an array whose size is allocatable and the name of that variable to a dimensional array is spin. Here I gave two colons which means it is a 2D array and its size shall be discussed when we implement, when we run the code, ok. So, here I am saying, please gave the number of lattice points in one-dimension.

So, that is L, you can give L to be 20, 30, 40, 5, 10 whatever you want. Number of iterations, that you want at that particular temperature and we shall first do this and see how thermodynamic quantities evolve. Once you have given L in the system you have to allocate this array say that the spin is, the spin variable the spin array is L cross L, then you set the initial energy to be instantaneous energy of the system is a basically defining variables energy to be 0, this is the energy of the system.

Now, this is not the thermodynamic energy, this is the instantaneous energy, of; when you generate a microstate with a particular configuration of the spins. What is the energy of that lattice, of that particular configuration of spins or that microstate? You have to average over different microstates to get the thermodynamic average, remember that, right. We shall do that later. So, E has been set to 0, the magnetization, magnetization, not the thermodynamic magnetization, the instantaneous magnetization.

What is the; if you sum over all the spins some are plus some are if you sum over all the spins, what is the total magnetic moment, what is the total value of the spin of the lattice that is what M is measuring; the instantaneous magnetization of the lattice, right, instantaneous value of spin of the lattice. Here I have defined N the total number of spins to be L cross L, L being the number of spins along one particular direction.

(Refer Slide Time: 10:53)



```
  N=L*L  ! Total number of spins in lattice.

  call random_seed

!Initialize your lattice
  open(71,file='initial_ising.dat')
  p=0
  do i=1,L
     do j=1,L
        call RANDOM_NUMBER(r)
        spin(j,i) = 1

!         if(r<0.5)then
!            spin(j,i)=-1
!         else
!            spin(j,i)=+1
!         end if
        ! WRITING DOWN INITIAL CONFIGURATION.
        write(71,fmt='(4g10.8)') float(i),float(j),float(p),float(spin(j,i))
     end do
  end do
 close(71)

!Calculate initial magnetization and energy
do i=1,L
   do j=1,L
      a=i+1;b=i-1;c=j+1;d=j-1  ! identifying the 4 neighbours of spin(j,i)

      if(i==L)a=1  ! PBC: periodic boundary conditions.
      if(i==1)b=L
      if(j==1)d=L
      if(j==L)c=1
      E=E-J_ising*float((spin(j,i))*(spin(a,j)+spin(b,j)+spin(i,c)+spin(i,d)))
      M=M+spin(i,j)
   end do
end do
```

Here we are calling a random number seed basically to generate a set of random numbers. And what is the first step in the flowchart? If you go back in the video and look at it, one has to initialize the lattice. What does initialize the lattice mean? Give the values of the spin plus 1 or minus 1 at each lattice site and that is exactly what I am doing.

So, here you see that do i equal to 1 to L, so the value of i changes from 1 to L, right. do j equal to 1 to L, so there were two indices x and y, corresponding to x and y, so i and j. So, each varies from 1 to L. Then you are generating a random number here, call random number r. The random number is stored in this variable r. And here I am choosing a random initial configuration which means that you go to each site in the lattice and with equal probability you assign the spin the spin at that lattice point the value of the spin at that lattice point to be either plus 1 or minus 1, right.

So, how do you do that? You basically generate a random number r and if that random number r is less than 0.5, so random number r it will be generated, I am using a uniform random number generator, which means that the random number generator will have uniform probability to be generated at any value between 0 and 1, right. So, the 50 percent of the time the random number will be less than half and half, if you generate another random number the probability that its value will be greater than 0.5 half is again half.

So, so if r is less than 0.5 then you say ok, give the value of the spin variable spin j, i. So, one particular lattice site and j and i are varying, do i equal to 1 to L, do j equal to 1 to L. So, if spin j, i here if this site is conditioned r is less than 0.5, this condition is satisfied then spin of j, i is given minus 1. If it is greater than 1, as I said equal probability of being either plus 1 or minus 1 then you give it plus 1. So, on an average half of the spins in the lattice will be pointing up, half of the spins on the lattice will be pointing down, right. So, we have assigned spin values to this.

You want to see whether you have done it right or not. For that what you do? You can write down i as you change the lattice site j and basically this why I have given this, usually understand later. I mean it is basically later for visualization, it is a dummy variable which I have defined to be 0. And float spin i, j I have whatever value has been assigned here or here I am writing it down. So, you can actually after you have initialized the lattice you can go to basically any lattice size which is specified by i and j, and check what the value of the spin has been given, right.

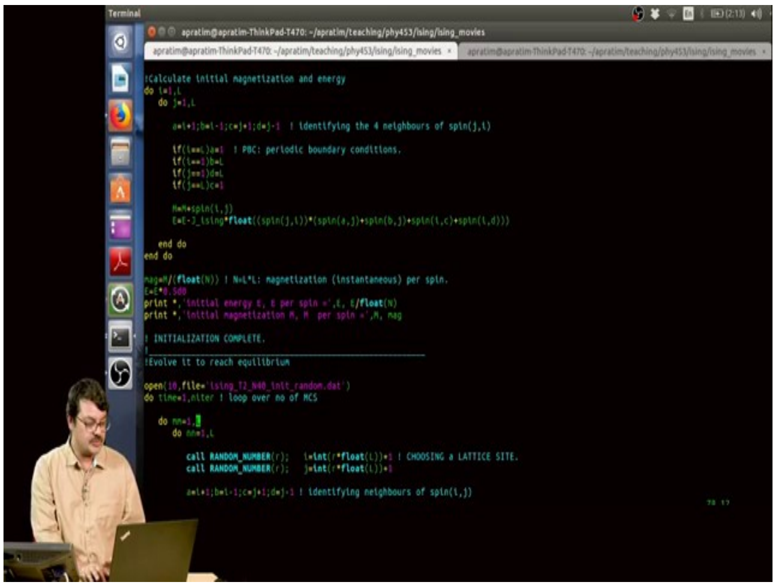
And that I am writing down in a file and 71 which has been initialized just before this loop. So, open files 71, file equal to initial ising dot da. Initialising dot dat means essentially that you are writing down the initial spin configuration of the lattice in this file just for a cross check whether you have initialized it appropriately or not. As I told you our thermodynamic properties are not going to depend upon the initial condition you have to evolve the system by suitable metropolis algorithm to equilibrium.

Also to choose initial configuration where all spins are pointing up or all spins are pointing down, initial configuration does not matter. Here I have simply chosen to give all spins to assign spins at different points in the lattice randomly, half pointing up, half pointing down, but if you were suppose wanted to basically say, I want to look at all spins pointing up. So, if you wanted to give all the spins of the lattice to be say plus 1, so that they are all pointing up or minus 1 if they are all pointing down, then what you can simply do is comment out this part and do this.

So, do i equal to 1 to L, do j equal to 1 to l. So, it systematically says the what the code will do when you run it. So, it will go I equal to 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 and corresponding to j equal to 1 and then j will also change. So, basically you are scanning over the entire length of the lattice through this and every lattice site, at every lattice site the spin variable is set to minus 1 here. You can equally do it plus 1, does not really matter. So, here then all spins are now pointing up. I am just saving this code by colon w, right. So, this is your initial initialization of the lattice.

Now, what do you do? You calculate the initial magnetization or the in value of the total number of spins of the system and the energy of the system.

(Refer Slide Time: 17:27)



```
!Calculate initial magnetization and energy
do i=1,L
   do j=1,L

      a=i+1;b=i-1;c=j+1;d=j-1  ! identifying the 4 neighbours of spin(j,i)

      if(i==L)a=1  ! PBC: periodic boundary conditions.
      if(i==1)b=L
      if(j==1)d=L
      if(j==L)c=1

      M=M+spin(i,j)
      E=E-J_ising*float((spin(j,i))*(spin(a,j)+spin(b,j)+spin(i,c)+spin(i,d)))

   end do
end do

mag=M/(float(N)) ! N=L*L: magnetization (instantaneous) per spin.
E=E*0.5d0
print *,'initial energy E, E per spin =',E, E/float(N)
print *,'initial magnetization M, M  per spin =',M, mag

! INITIALIZATION COMPLETE.
!
!-----------------------------------------------------------------
!Evolve it to reach equilibrium

open(10,file='ising_T2_N40_init_random.dat')
do time=1,niter ! loop over no of MCS

   do nn=1,L
      do nn=1,L

         call RANDOM_NUMBER(r);   i=int(r*float(L))+1 ! CHOOSING a LATTICE SITE.
         call RANDOM_NUMBER(r);   j=int(r*float(L))+1

         a=i+1;b=i-1;c=j+1;d=j-1 ! identifying neighbours of spin(i,j)
```

So, to calculate the total energy and magnetization of the lattice the initial one, so what we do is again do i equal to 1 to L do j equal to 1 to L which means that you are essentially going to each lattice site you are scanning over each lattice site the spin at each lattice site and here we are identifying the neighbours. So, basically if you are. So, suppose you have a spin whose location is spin i comma j. So, a equal to i plus 1 which is basically the right hand side of your this spin on the right hand side of your target spin i, j, b equal to i minus 1 which means on the left, c equal to j plus 1 which is basically on the top of your lattice site i, j and d equal to j minus 1 which is at the bottom of the lattice site, right.

And now, basically if by chance i is equal to L which is basically if the spin under consideration is at the boundaries then its right hand neighbour will be basically the other side of the lattice by periodic boundary conditions. So, that is why i have written if I equal to L, then a, which is the neighbour on the, right hand side is essentially equal to 1. Similarly, if I equal to 1 which is on the left boundary if the spin in under consideration is on the left boundary then the neighbour which is to the left of this one, so on this side is equal to L. It is basically periodic boundary conditions which we discussed in the past. And similarly these two conditions if j equal to 1 which is at the top boundary of the lattice then the neighbouring spin on the top of it is L and so on so forth. This is on the other side.

Now, you have basically identified the 4 neighbours of spin i, j even if i or j lies somewhere on the boundary periodic boundary conditions have been invoked to have correctly identified the 4 neighbours of any spin on the lattice i, j, k, L wherever being it on the lattice. And M, which had been set equal to 0 before the beginning of the simulation, I am saying M equal to M plus spin i, j.

So, the value of the variables M you are adding the value of the spin at a particular site and you are going to each lattice site and adding the value of the spin at that lattice site. If all the spins are pointing up then basically at once you have scanned with the entire lattice you are going to add one, every time you visit every lattice site and then the end of the loop here you will have the value of M to be n into 1, 1 being because all spins are up.

However, if you started with a random initial conditions then some of the spins will be up and some of the spins will be down and the value of this M, the variable M, will be you are adding some times plus 1 to it, sometimes minus 1 to it with equal probability nearly. So, it the value of M after you have scan through the lattice will be some number close to 0, ok. Similarly, you can calculate the value of the energy of the lattice, so you have been going to each lattice site E had also been set to 0, right at the beginning of the code.

And now, you are doing minus jising, jising being j the energy scale of the problem. And then spin j,i. So, you have to do the multiplication S i S j sum over all neighbours and that is exactly what has been done here. Spin a, j; spin b, j and the two neighbours on the left and on the right of this spin i, j and this is the spin on the top and at the bottom, bottom neighbour of spin j, i. So, I have multiplied spin j, i by the spin values of the spin at the lattice sites on the top, bottom, left and right and by that you get essentially the value of this energy for a particular spin j, i.

Then of course, you go to the next lattice site and again identify its neighbours, calculate the energy of the interaction and you add it to E. So, I am saying add though there is the minus sign here because the Hamiltonian itself comes with minus j summation S i dot S j and that is all that I am implementing here. Here, what is done mag. Mag is essentially the magnetization per spin. When I am say magnetization I mean the instantaneous magnetization.

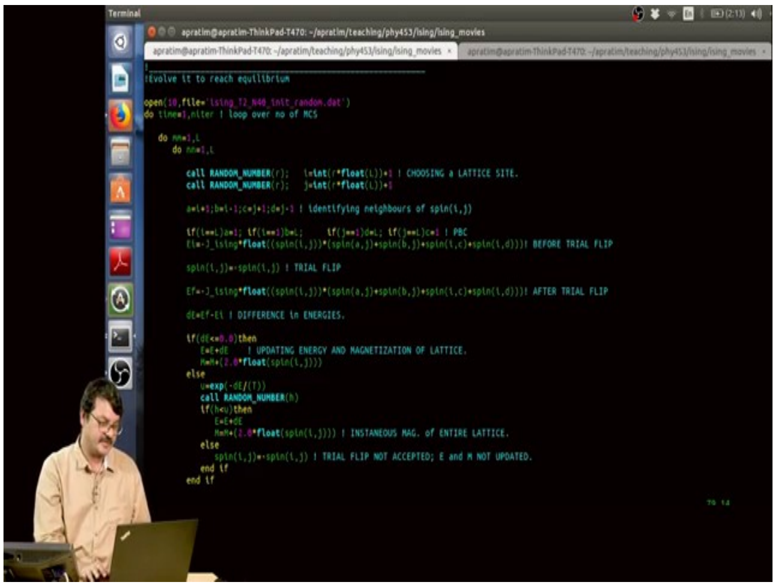
The magnetization of a particular microstate, the value of the magnetic moment of a particular microstate, right; so, that is M divided by float N, N being the total number of spins in the lattice. That is trivial, but what is not so trivial is the value of the energy. So, here I have multiplied the value of the energy of the microstate of that spin configuration by half. Why this half? Because as I go from each lattice site 1, 2, 3, 4 and so on and so forth, I calculate the energy of interaction between 1 and 2.

And then when I go to the next lattice site I again calculate the energy of interaction between 2 and 1, because if something is , right neighbour, if 2 is the right neighbour of spin 1 say, then spin 1 is the left neighbour of spin 2. So, every energy has been double counted in this calculation. If you just think about it you will figure it out. And since two basically correct for this double counting I have multiplied it by 0.5.

Then, what am I doing? I am saying write down the total initial energy and energy per spin on the screen and we shall see that when we run the code. Here I have said write down the initial magnetization, initial magnetic moment not instantaneous magnetization of the entire lattice and M per spin. So, I have just written down M mag E and E by float N. So, this is initialization complete.

You have already access to all the spins which have been initialized on a lattice. Moreover, we have calculated the initial energy of the entire lattice for that microstate. We have calculated the initial magnetic moment or the spin, the total spin of the lattice at time T equal to 0 before the simulation starts. And you have also access to the magnetic the spin per lattice site and the energy per lattice site. So, the next thing to do is basically evolve it to equilibrium.

(Refer Slide Time: 25:01)



```
!Evolve it to reach equilibrium

open(10,file='ising_T2_N40_init_random.dat')
do time=1,niter ! loop over no of MCS

   do mm=1,L
      do nn=1,L

         call RANDOM_NUMBER(r);   i=int(r*float(L))+1 ! CHOOSING a LATTICE SITE.
         call RANDOM_NUMBER(r);   j=int(r*float(L))+1

         a=i+1;b=i-1;c=j+1;d=j-1 ! identifying neighbours of spin(i,j)

         if(i==L)a=1; if(i==1)b=L;     if(j==1)d=L; if(j==L)c=1 ! PBC
         Ei=-J_ising*float((spin(i,j))*(spin(a,j)+spin(b,j)+spin(i,c)+spin(i,d)))! BEFORE TRIAL FLIP

         spin(i,j)=-spin(i,j) ! TRIAL FLIP

         Ef=-J_ising*float((spin(i,j))*(spin(a,j)+spin(b,j)+spin(i,c)+spin(i,d)))! AFTER TRIAL FLIP

         dE=Ef-Ei ! DIFFERENCE in ENERGIES.

         if(dE<=0.0)then
            E=E+dE    ! UPDATING ENERGY AND MAGNETIZATION OF LATTICE.
            M=M+(2.0*float(spin(i,j)))
         else
            u=exp(-dE/(T))
            call RANDOM_NUMBER(h)
            if(h<u)then
               E=E+dE
               M=M+(2.0*float(spin(i,j))) ! INSTANEOUS MAG. of ENTIRE LATTICE.
            else
               spin(i,j)=-spin(i,j) ! TRIAL FLIP NOT ACCEPTED; E and M NOT UPDATED.
            end if
         end if
```

For that you do start Monte Carlo steps and here you see that I have a loop which says time equal to 1 comma niter. Time is a dummy variable you can call it by any other variable. We are essentially doing ensemble average, but we are generating one microstate after the other using the metropolis algorithm which is implemented below. And we are saying we are going to have a niter, niter number of iterations, niter stands for number of iterations and niter.

We are going to have n niter number of independent microstates that is what we are going to generate after every Monte Carlo step, ok. So, we are actually going to generate more, but. Now, do mm equal to 1 to L, do nn equal to 1 to L. These are mm and nn are dummy variables. We have basically going over each site in the lattice. But we are not going through each site in the lattice in a sequential manner. Why and how? Well the how first.

So, we are generating a random number here r and another random number r, we are multiplying it by the length of the lattice float L. So, basically what you have i here is I am choosing random lattice site. So, generating a random number between 0 and 1, multiplying it by L, so you have a random number between 0 and L. They will take the integer and by doing a plus 1 and you should think why I am adding a plus 1 or if you delete it what happens. Basically, choosing a random coordinate in the x direction and here I am choosing a random coordinate in the y direction. So, I am choosing a random lattice site i and j.

We are identifying the neighbours just as we did previously. Here we are again implementing the periodic boundary condition in case i and j is a set on the boundaries of the lattice, and here we are calculating the initial energy for spin i, j for a particular value of spin. And the formula is exactly the same as previously except we were adding the energies previously here I am just looking at the energy of a particular spin i, j and the energy with its neighbours, right.

Now, as we said in the metropolis algorithm we give a trial flip. So, if the spin was pointing plus you say let it spin be minus 1, and if it is minus 1 then you gave it a trial flip of plus 1. So, here I have given a minus sign, whatever be the value of spin i, j I have given a minus sign to it. And with this changed flip and the changed value of the spin, after the spin flip I am calculating the new value of the energy of that configuration.

So, I am talking about a spin i, j and its 4 neighbours, right. So, Ei stands for initial energy, Ef for final energy. After the trial spin flip. And you here you calculate the difference in energies dE equal to Ef minus Ei. Now, as for the metropolis algorithm as we discussed if dE this change in energy due to the trial flip if it is less than 0, if energy decreases then basically the trial spin flip is accepted with probability 1.

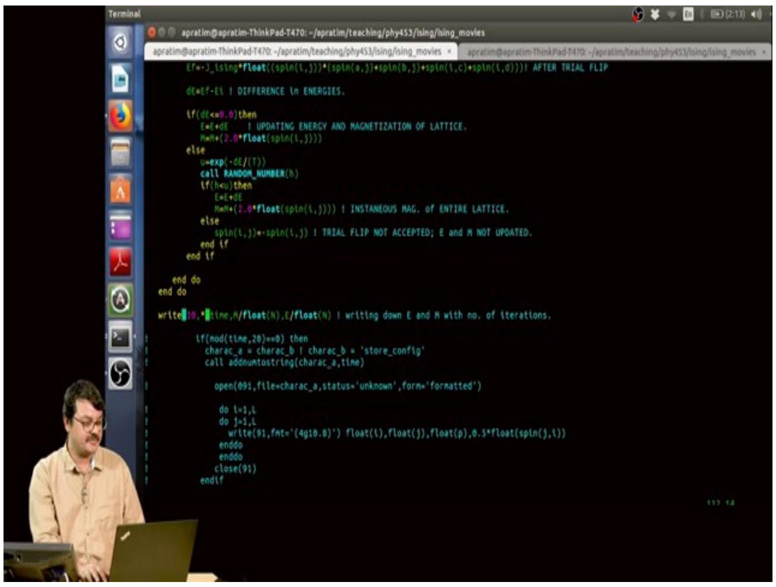
So, if, so here I have already done the trial spin flip if dE is less than 0, the change in energy is negative then you update the total energy of the system. The total energy of the system was E the difference in energy was dE, so I have written the new energy after this trial spin flip which has been accepted , the new energy is E plus dE, and the new magnetization of the entire lattice M was the instantaneous magnetization of the entire lattice, that is M plus 2 float spin i, j, right, plus 1 goes to minus 1. So, the change in spin of the lattice is plus 1 minus minus 1, so that is why 2, right. So, that is how you get this.

But, so basically you have accepted the spin flip here in these two steps and updated the total energy of the lattice and the total instantaneous spin state of the lattice. But if dE was greater than 0 or. So, here just notice that this is less than equal to 0, but if dE was greater than 0. So, this is this ‘else’ condition for that, then you are supposed to accept the spin flip with probability delta E by K B T. So, how do we accept spin flip with probability E to the power delta E by K B T?

If you remember Prasenjit’s lecture about cumulative distribution I am basically and about random numbers and so on so forth, I am using that idea. So, here I calculate u, u being a dummy variable is exponential minus del E by K B T, K B has been set to 1. So, basically here what ideally it should be is K B T, but I do not need an extra variable K B which has anyway be set to 1. So, I have just written dE by T. Call a random number h, if h is less than u, this is what is u, that is E to the or delta E by K B T, then accept the spin flip, then you again update the value of E and the value of M just as was done in these two steps. And the spin is already been trial flipped here, so then that step is accepted.

However, if the if this condition if this condition is not satisfied, if this random number is greater than u, then this spin flip is not accepted I put the spin I, j to be minus spin I, j. So, suppose it was a positive spin I had given a trial spin a flip to be minus 1. Now, I say, this trial spin flip is not accepted then I have to put it back to plus 1. So, that is exactly what I am doing in this step, ok. And of course, I do not need to update the M and the E. Now, this is being done, this.

(Refer Slide Time: 32:36)


Terminal
apratim@apratim-ThinkPad-T470: ~/apratim/teaching/phy453/ising/ising_movies

```
        Ef=-J_ising*float((spin(i,j))*(spin(a,j)+spin(b,j)+spin(i,c)+spin(i,d)))! AFTER TRIAL FLIP

        dE=Ef-Ei ! DIFFERENCE in ENERGIES.

        if(dE<=0.0)then
           E=E+dE    ! UPDATING ENERGY AND MAGNETIZATION OF LATTICE.
           M=M+(2.0*float(spin(i,j)))
        else
           u=exp(-dE/(T))
           call RANDOM_NUMBER(h)
           if(h<u)then
              E=E+dE
              M=M+(2.0*float(spin(i,j))) ! INSTANEOUS MAG. of ENTIRE LATTICE.
           else
              spin(i,j)=-spin(i,j) ! TRIAL FLIP NOT ACCEPTED; E and M NOT UPDATED.
           end if
        end if

     end do
  end do

  write(10,*)time,M/float(N),E/float(N) ! writing down E and M with no. of iterations.

!          if(mod(time,20)==0) then
!            charac_a = charac_b ! charac_b = 'store_config'
!            call addnumtostring(charac_a,time)
!
!              open(091,file=charac_a,status='unknown',form='formatted')
!
!              do i=1,L
!              do j=1,L
!                write(91,fmt='(4g10.8)') float(i),float(j),float(p),0.5*float(spin(j,i))
!              enddo
!              enddo
!            close(91)
!          endif
```

So, there are now n attempts n being L cross L attempts which are being done. So, we are randomly choosing spin n times, trying to flip the spin, sometimes it is accepted, sometimes it is not depending upon the value of delta E and specially K B T, the thermal energy. And at the end once you have done it for essentially n times you have completed one Monte Carlo step. These to ‘end do’s correspond to these two end dos, right. And it has been written in a manner, so that these end dos and end ifs can easily be identified. This is the way you write programs, so that you do not get lost between end dos after hundred lines and dos and end dos, ifs and end ifs after 100 lines. So, that is why you basically structure the writing in the way it has been done here.

And after one Monte Carlo step that is n attempts you write down the total magnetization instantaneous magnetization per spin and the energy of the system per spin in a file essentially, right. So, this file is file number 10 and it had been opened right at the beginning here; so, ising T 2 N40 and it random. So, this says that the temperature is 2, the lattice size is 40 actually it should be L not N. So, it is a 40 cross 40 lattice and with initial random condition, right.

So, now, if you do this niter times then you have this loop was here. Loop over Monte Carlo step and that loop ends here, and these, then you would have got the energy and magnetization, you would have written it down after every Monte Carlos steps niter times and you can certain to be 1000 or 1 lakh or 1 million it is your choice. A question: so, just to remind you that the spin to be updated is chosen randomly in the lattice. In one Monte Carlo steps you are doing n attempts to spin flip, but the spin to be chosen to be given a trial flip is chosen randomly.

It is a consequence, in one Monte Carlo step it is quite possible that 1 or 2 of the spins or 5 of the spins, you I have given trial flips more than once. On the other hand, a few particular spins have not been given at all a trial flip which is possible because you are choosing the spins randomly. But in the next or the next Monte Carlo step the ones which have been left out to give a trials flip you might choose that twice and the ones which basically you attempted spin flip twice or thrice in one Monte Carlo step they will be basically this time left out and the ones which were previously left out they might have two attempts.

Now, if you have many Monte Carlo steps on an average and if you have a good random number, then on the average all spins the number of attempts to flip a spin will be accepted equally, ok. If you have just 5 Monte Carlo steps, of course, some spins are not going to be updated, some spins are going to be updated more than once or at least try attempt to make a spin flip will be more than 1, but if you do it over say 10,000 or 1 million Monte Carlo steps then on an average each spin will have an equal attempt, each will have a equal number of trials spin flips.

Now, whether those trial spin flips are accepted or not that will of course, depend upon the temperature. How? Well, to do that we have to actually run the code and see, as we change the temperature as we change the lattice site what happens and that is exactly what we are going to start doing in the next class or tutorial.

Thanks.