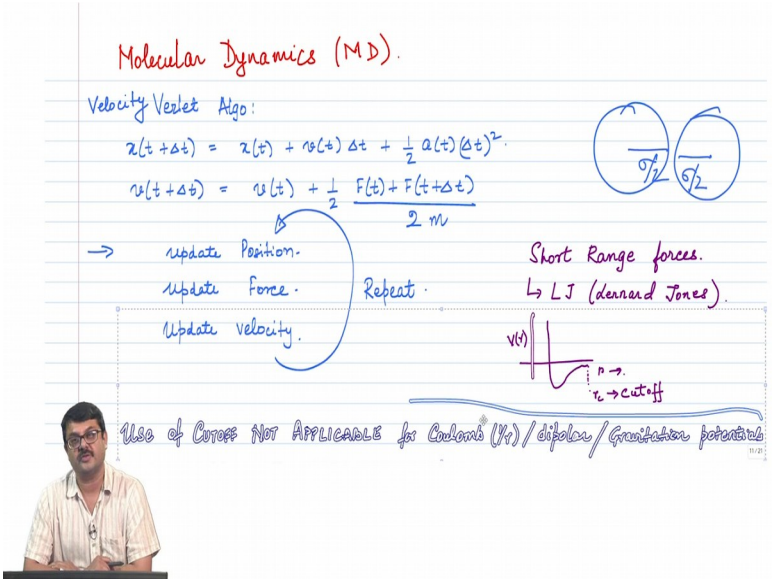
Computational Physics
Dr. Apratim Chatterji
Dr. Prasenjit Ghosh
Department of Physics
Indian Institute of Science Education and Research, Pune

Lecture – 48
Molecular Dynamics Details and Algorithm Part 01

(Refer Slide Time: 00:17)



So, welcome back to the second class of the Molecular Dynamics Module. So, we shall just continue from where we have talk in the last class. So, it is in short Molecular Dynamics in short is also called MD and in the last class we were discussing velocity verlet algorithm right.

And the basic message is in a molecular dynamics of my simulations, you essentially update position force and then velocity one after the other and you keep repeating this over many iterations. So, that the position of n particles which are in the simulation box and the velocity of these n particles keep on changing. The particles move around in space and they collide with each other the bounce of each other, they move in different directions, they change positions the change velocities and once equilibrium is reached one can calculate the time averaged of relevant statistical or thermodynamic quantities.

How does one update position? It is simply x the position at t plus delta t depends upon the position of the previous time t and without the delta t. The velocity at the previous

time into d t plus half a for acceleration delta t whole square right. And how do you update the velocity as we discussed last time the velocity at time t plus delta t depends upon the velocity at the previous time t.

But interestingly it depends upon the force both at the previous time t and the current the present time t plus delta t. So, what one has to do is update the position like in this step and then update the force as per the new position. So, that one gets F plus t plus delta t and then update the velocity and this is repeated in this order.

We also discussed in the last class, that the update of position and update of velocity is relatively very simple you just use these two formulas. Whereas, the update of force is computationally rather expensive. So, a large part of the discussion shall actually be about discussion of how to update the force between interacting particles in an efficient computationally efficient manner.

For the example that we chose we were discussing short range forces in particular the Lennard Jones force and the Lennard Jones potential schematically is drawn here. So, this is just a recap of this we have all discussed this last time, but anyway this is a recap quickly 5 minute recap and so the Lennard Jones potential and we had written down the form of the potential in the last class schematically looks a bit like this.

So, there is a at when the distance between two particles is very small, small is defined to be smaller than the diameter then there is an extremely repulsive force between the particles. And at slightly shorter distances there is an attraction and this weak attraction goes off to 0 as you go to larger and larger distances.

And if you remember we had decided to use a cutoff of the potential, because the potential is the interaction potential the interaction force is extremely weak at distances larger than around 2.5 or 3 sigma. What is sigma? Sigma was the diameter of the particles or rather if basically you had two particles like this, this is sigma by 2 this is sigma by 2 right. The radius of the two particles and when there were two particles just the distance between the centre of the two particles is sigma.

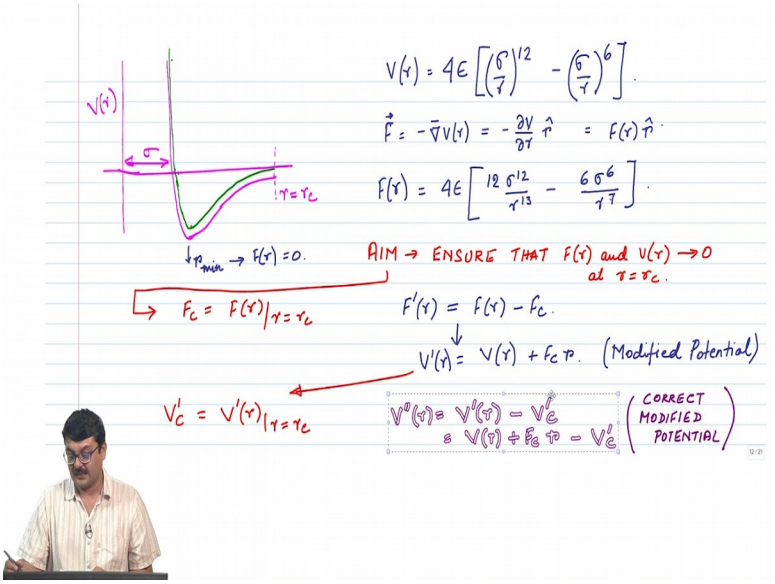
So, just when the distance of the two particles is just beyond sigma there is an attractive force, but at distances of three sigma the attractive force or the interactive interaction is extremely weak and hence it is not worth computing the forces at distances greater than 3

sigma say or 2.5 sigma. You can choose your cutoff that modifies the potential slightly if you cut it off, if you set the potential and thereby the force to 0 at greater than sigma at distances of greater than 3 sigma, but it does not change the physics.

We could use a cutoff since this was a short range force the expression of the force. If you remember or rather the expression of the potential the Lennard Jones potential was 4 epsilon was a measure of the strength of the interaction, the basically the depth of this potential into sigma by r to the power 12 minus sigma by r to the power 6, so there is a sharply changing force.

However if you had to work with interactions such as the coulomb or the dipolar or the gravitational interaction which were basically the v of r goes as either 1 by r ah. For coulomb and gravitation and for dipole it goes as 1 by r square, those are long range forces and one cannot use the concept of a cutoff. As I told you as we discussed even last time they one use this slightly different methods to calculate the forces, the rest of molecular dynamics simulations remains the same.

(Refer Slide Time: 06:19)



So, in the last class if you remember we were discussing, that if we cutoff the potential here at r equal to r c, what you would get is a discontinuity of the potential here. But you could then shift the potential, so that it goes smoothly to 0. But still the force does not go smoothly to 0 and what the new topic that we are going to start with today is how to get the forces how to do the calculation. So, that the force the modified force goes to 0

smoothly at r equal to r c as well as the potential of interaction between two particles go smoothly to 0 at r equal to r c., rc be the distance where the potential goes I mean it is set to 0 beyond that it is the cutoff distance, so called right which you can choose to be 2.5 or 3 sigma.

So, this is the expression of the Lennard Jones potential 4 epsilon sigma by r to the power 12 minus sigma by r to the power 6. And then the expression for the force and forces a vector quantity and hence there is vector sign on top of the force is nothing but minus grad v of r. And since this is not a theta dependent or a phi dependent potential the expression for the force is just minus del v del r into r cap; r cap being the unit radial vector which we are calculating of course, I have written the expression in the spherical coordinates.

And so that can be written as the magnitude of the force F of r into r cap. What is F of r you have take a simple derivative with respect to r and that is nothing but 4 epsilon this remains the same and you were to take a derivative of r or rather r to the power minus 12. But remember the expression for the force comes with an extra minus sign here right and so when you take the derivative you would get 12 into sigma to the power 12 upon r to the power 13 minus 6 sigma to the power 6 upon r to the power 7.

You can just work it out and it is no big deal and you have to ensure that this expression of F of r, this expression of F of r goes smoothly to 0 at the cutoff and remains 0 beyond for distances for r greater than the cutoff distance and the potential also remains smoothly to 0.

So, how does one do that so this is our aim to ensure that both F of r and v of r goes smoothly to 0 at r equal to r c and the way one does it is calculate F of c calculate F of c. So, or rather F c which is the value of F of r; the value of F of r at r equal to r c, so that I denote by the symbol F of F c.

Now, if you have a slightly modified force expression of the force F dash r, which is nothing but F of r this F of r this F of r right minus F c. Then basically what you have is that the force has finite values. So, here is a schematic expression for the potential I have not drawn the force, but remember that v of r at r min is equal to basically here the slope is 0. Basically it changes sign it goes from positive slope to negative slope. So, in

between it has to be 0, which means the force has to be 0 at this point at this point of the simulation right.

And you can similarly draw you can similarly draw this potential as well schematically you should try it out and the message is whatever the value of F c b at r equal to r c you are setting that to be 0. So, it is basically going to smoothly to go to 0 at r equal to r c, because they are shifting the potential the same ideas that we discussed in last class for V of r. So, we are shifting the force and after and at distances greater than r equal to r c it is anyway remaining 0 right.

So, that is how you modify it force expression for the force and you can check that the value of F c if you choose r c to be around 3 sigma c is very small right. But if you have this F dash r to be the operating expression for force, the slightly modified expression for the force. What you have landed up is you have practically also modified the expression for the potential, the potential is no longer V of r there is no longer V of r as in this. Instead you have a modified potential V dash r which is if we take do the proper integration is V of r plus F c r.

Now, if you take the derivative of this quantity you will get this expression for the force, which we have chosen to be your operative which I am suggesting that you choose as the operative definition, the slightly modified expression for the force instead of just this F of r right.

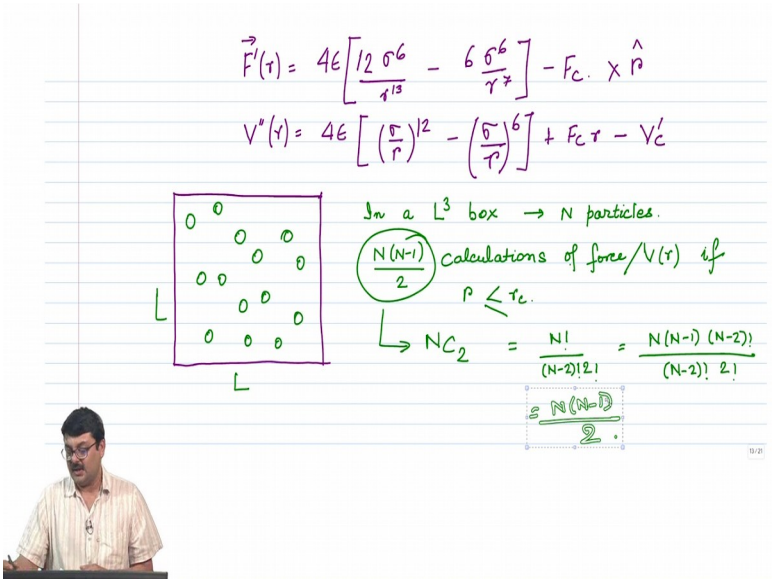
So, your modified potential now is V dash r which is V of r plus F c r. But this expression for V dash r the modified potential V Vdash here the dash is not derivative right, this is just V dash stands for modified potential. So, this one does not go to go to 0 smoothly at r equal to r c, what do you do how do you make this expression now go smoothly to 0 you calculate V dash c subscript c. What is it is nothing but the expression of V dash r V dash r right, at r equal to r c and you choose your new corrected modified potential to be V double dash r which is V dash r minus V dash c.

And now this quantity which is basically V of r plus F c r minus V dash c, this quantity goes very smoothly to 0 at r equal to r c and remains 0 beyond r equal to r c. Moreover if you use this expression for the potential the corresponding force expression for the force is this and this in turn also goes smoothly to 0 at r equal to r c and remains 0 for r greater than r c and that is exactly what we wanted. So, the message is to make the force and

potential both go smoothly to 0, so that we have relatively efficient calculation of the force.

We not only; so what we do is not start from the potential set it to 0 and then calculate the force, but first calculate the force set it ensure that it goes smoothly to 0 work out the what is the modified potential which is V dash r in this case and then from that ensure that that quantity goes smoothly to 0 at r equal to r c and remains. So, that is how you do it right and that is the correct modified potential.

(Refer Slide Time: 14:47)



So, your actual operating potential is now a modified Lennard Jones potential, modified very slightly as you should check as we discussed last time that the value of F c is very small and the value of V dash c is also going to be very small. But anyway technically you have modified it. It has been checked that that does not change the physics behavior much.

But of course if you choose r c equal to say 2 or 1.5, of course we have changed the potential drastically you would have modified the depth of the potential and it will give you different physics. But if you choose r c to be 3 sigma you practically are working with a as good as the original Lennard Jones system.

So, these are the two finally operating expression for the force F dash r and V double dash r and these are the expressions here you have F c. And now if you basically

calculate the force and make the particles move around by these two expressions of the force and the potential, you would see that there is no discontinuity the energy is perfectly conserved.

There is no kicks some random unphysical kicks which appear, as the two particles move away farther away from each other and the distance becomes greater than r c. Or alternatively if two particles because of the virtue of the previous velocities they approach each other and they and the distance between them becomes less than r c. Even if such things happen the energy will be conserved as it should be assuming you are doing an N v t ensemble ok.

In whatever we have discussed along this teapot is has will be discussed a bit later, till now whatever we have discussed is more like a N v e ensemble where the energy is conserved. But later we will have to discuss t also how to set the temperature it will take some time. And so pictorially what we have is essentially a box maybe a cubic box, but you can also choose a non cubic box for the case of simplicity we will take a cubic box here I have choose shown L cross L.

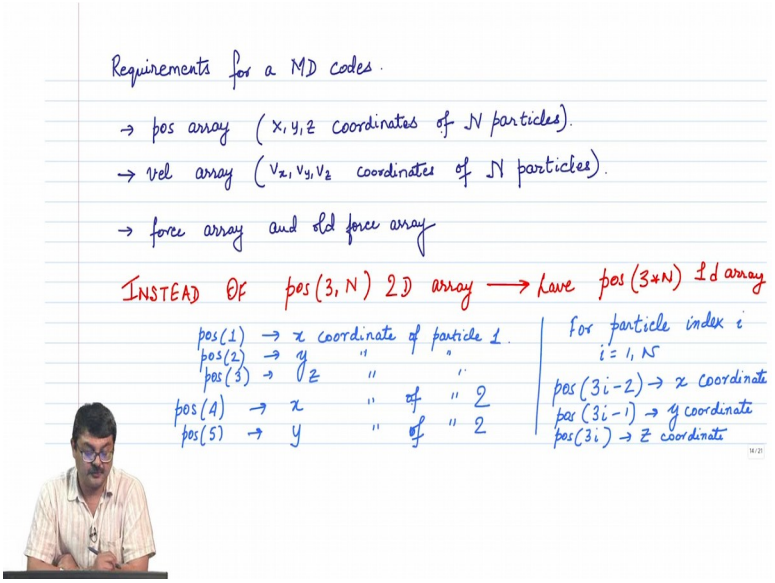
But if you are doing a simulation in 3D you will have l cube box in which there will be N particles and taking every pair of particles you have N into N minus 1 by 2 calculation of the force and the potential and you calculate this force and the potential only if r is less than r c right.

Because if r is greater than r c then essentially the potential is 0 the force is 0 and only if r is less than equal to r c; less than equal to r c, do you need to calculate the force. So, you have already saved some amount of the calculation by taking a cut off, later we will discuss how we even capitalized much more by using neighbor lists. But that has to wait for some time.

So, I hope you understood why I said that there will be N into N minus 1 by 2 calculations it basically comes from NC 2. So, there is a pair by is interaction every particle is interacting with we are the particle even if it is interaction be 0, then you do not need to calculate the expression for the force. And there are NC 2 C being combinatorial symbols NC 2 ways choosing a pair of particles from N particles and NC 2 is nothing by N.

But N factorial upon N minus 2 factorial upon 2 factorial and that you can easily check is nothing but N into N minus 1 by 2 right. So, when N is large you could say that this is an N square process ah, but at least in some cases because you have a cutoff you do not need to calculate the expression for the force and the potential.

(Refer Slide Time: 19:11)



So, this is the broad picture of what is needed in a molecular dynamics or a MD simulations. Now, let us go slightly into the gory details and discuss the algorithm in greater detail and that is exactly what we did for the Ising model as well. If you remember we first discussed the physics of the problem, then we discussed a bit of the how it is going to be implemented and then we will slowly went in greater detail. About how that algorithm is going to be implemented what various things does one have to be careful about and that is exactly the method or the path we are going to choose for our MD.

So, we have to talked in general about the principle of what the MD tries to do and now we are slowly moving about how to implement it in greater and gory detail. So, let us discuss what does one require essentially to write an MD code well one needs some big arrays, big especially if the number of particles you are simulating is large. So, you need a position array where you need the x y z coordinates of the n particles, you need the vel array call it vel velocity whatever you want to call it.

But I like to call it vel array which stores the v x v y and v z coordinates the v x v y and v z coordinates of N particles. Moreover one needs the force array they are basically the force total force acting on any particle will be saved, it could be 0 or it could be acted upon by multiple particles and you need to know the component of force F x F y F z of the force acting.

So, basically if you have even r hat vector you have to decompose into x hat and plus y hat and z hat. So, you need the force array and moreover you need the force at the previous time, because I just saw then when you update velocity you not only need the expression or the values of the force at that instant of time. But also the values of the force at the previous instant of time. So, you need a force array and then old force array.

Now, here is something which I strongly recommend, the reasons for this recommendation will be clear later when I discuss very briefly. What is the computer architecture and how different elements of the array is accessed by the CPU, how it is taken from the ram and how is access for the CPU but that is a separate ten fifteen minutes discussion.

At the moment you just accept what I say that instead of choosing these arrays pos vel and force to be a three cross n dimensional array three being. Because there are three degrees of freedom for each particle three degrees of position and three degrees of velocity and N is the total number of particles. So, instead of choosing a 2D array you should instead choose the length of the array you should first of all have a one dimensional array and the length of the array should be 3 into N ok.

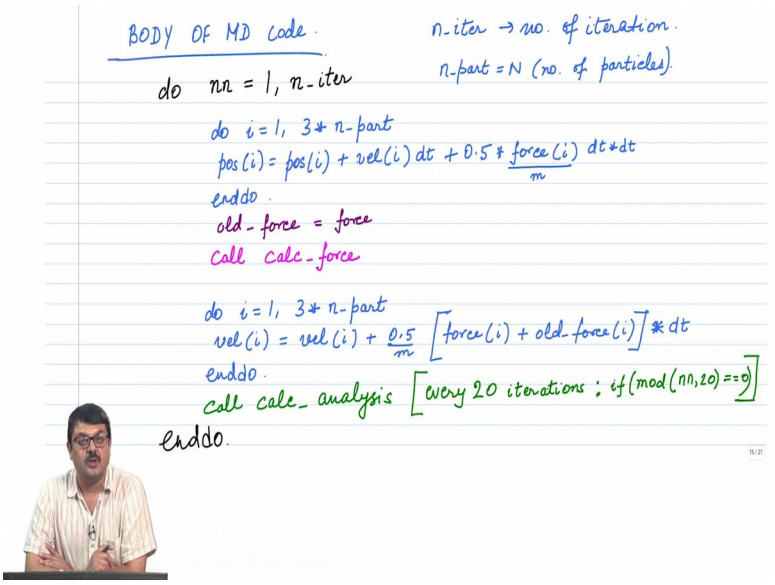
So, what does that mean? That do not have a three into 3 cross N two dimensional array, we should not have some anything like this. But instead have a array of length 3 into N it is one dimensional array, such that the pos 1. The first entry of the pos array will essentially store the x coordinate of particle 1 ok, pos 2 will store the y coordinate of particle 1 and pos 3 will store the z coordinate of particle 1, pos 4 will choose the x coordinate of particle 2, pos 5 will choose will store the y coordinate of particle two and so on so forth.

In general if you have n particles right the and suppose you are interested to update the position of the ith particle. So, i runs from 1 to N. So, then pos 3i minus 2 will store the x

coordinate of the ith particle 3 i minus 1 will store the y coordinate of the ith particle and pos 3 i will store the z coordinate of the ith particle.

So, when i equal to 1 3 i is basically 3 it stores the z coordinate of the first particle that is a quick cross check, you can similarly check when i equal to 2 and so on so forth. There are certain advantages basically the speed increases as i of the code increases while it is running and especially if you have large array number of particles 50000 or 1 lakh you have to run it for 1 crore iterations say 10 to the power 7 iterations it does make a difference.

(Refer Slide Time: 24:55)



So, with that what would be the main body of the MD code, we have of course discussed that update position update force update velocity and this goes on and on endlessly and more the number of iterations. Now, the body of the MD code will look something like this ,where you have do nn; nn is a dummy variable it is an integer variable whose value changes from 1 to n iter. What is n iter? n iter is the number of iterations right.

So, as then n keeps on changing you are basically progressing n time till the end of the end of the run. So, do nn equal to 1 to n iter and this of course, is this loop is finished here do and do I have basically kept all the statements which are within this loop this and this loop on the right, so this is called annotation. So, that the code becomes easier to read and then in each iteration for each value of nn 1 2 3 4 1000 and 10000 and so on so

forth. What is being done? This part is nothing but the update position do i equal to 1 3 into n part. What is n part? n part is number of particles right.

I choose n part n number n being number of particles and so basically you have if you have n particles you have to update the position x y and z, so 3 into n part. So, pos i equal to pos i plus vel i d t velocity of i, i runs from 1 to 3 n into d t into 0.5 half, this is nothing but 0.5 is half acceleration into d t square right. That is a formula which you know from your class 10 right.

At this part you have essentially updated the position both x y and z of n particles, then what do you do you already have the previous expression for the force that you save in an array called old force. And if you are doing force an 90 you do not have to run a loop if force is an array and old force is an array you can just directly write it like this and each element of the force will be saved into old force right.

Now, that the older values of the force have been saved you calculate the new values of the force, which will use this updated position right. So, you have to call calc force and of course we are going to discuss this in greater and greater detail and we have already discussed a bit.

And end is again you have a loop, where i goes from one to three n part exactly as in here right. Because you have v x v y and v z and you are essentially updating the expression of the velocity and there it is nothing but vel i into half 0.5 by M. A force i the ith particle and the old force of the ith particle into d t right. You have you have basically averaging over the current force at this time and the previous force.

So, at the end of this step you have essentially updated your position, your force and your velocity. Now, if you want you can collect data for future analysis like meant calculate the stress acting, the pressure which depends upon both force expression of the force and the velocity.

So, can basically call a subroutine to do whatever your analysis says and typically you do not call it every direction you call it every 20 or 50 or 100 iterations. And why is that because you want statistically independent you want to collect your data, whatever you want to analyze once the basically the microstates become statistically independent.

You know if you are integrating the Newton's equations of motion in one iteration, the particles are not going to significantly change it is position they are not going to significantly change it is velocity it is only up to 20 or 100 or 1000 iterations depending upon the value of d t right. This d t there is how small you have chosen the value you know with which you are going to integrate, that it is going to be decided. Over how many iterations will essentially get an independent microstate right, these discussions which all we also did for the Ising model if you remember ok.

So, how so suppose you want to calculate you want to call your analysis subroutine, where you collect data to calculate thermodynamic quantities every 20 iterations. Then you are going to call this calc analysis calculation of analysis every 20 iterations. And how do you implement that you say that I am going to call this subroutine only if mod nn 20 equal to 0. So, just look up what is what does the mod do so only when nn mod nn 20. So, if nn is a multiple of 20 then the remain the mod nn 20 will be equal to 0, so every and if you put some other number 100.

So, basically mod nn comma 20 gives the remainder right. If you divide nn the first number by 20 the nn by 20, but it gives the remainder. And only if it is equal to 0 which means nn is exactly divisible by 20 and if that condition is satisfied. That means, that nn is a multiple of 20 you are going to call, if that condition is satisfied then call Calc analysis. Which means every 20 iterations or if you instead of 20 you put 100 here, then every hundred iterations it is going to call this subroutine right.