Advanced Thermodynamics and Molecular Simulations

Prof. Prateek Kumar Jha
Department of Chemical Engineering
Indian Institute of Technology, Roorkee

Lecture - 45
Practical Aspects of Molecular Simulations

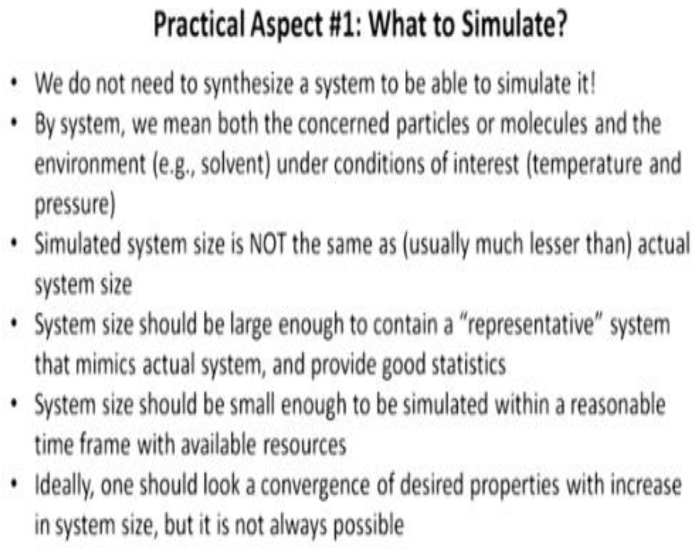
Hello all of you. So, in today's lecture we will start talking about the molecular dynamics for MD algorithm, but before that let us quickly recap the practical aspects of the particle simulations that we have been discussing. So, the first thing in any particle simulation is what exactly should we simulate and the idea of the particle simulation especially Monte Carlo and Molecular Dynamics is that whatever we are simulating need not be physically synthesized. I mean, the point I am trying to make here is that we should not synthesize that particular system predicts some properties and then try to model it, instead we can synthesize the systems on a computer we can simulate it and predict properties and then we can decide whether we want to actually make it or not.

This kind of a paradigm is like called the first principle kind of a paradigm although Monte Carlo or Molecular Dynamics is not quite first principle because there is force field information that is coming from quantum chemistry or experimental data, but nonetheless once we know the force field of a given molecule we can look various solutions of that particular molecule various combinations of that molecules with others and all those kind of systems we do not have to synthesize to know what the behavior is like we can pretty much use our computing tool to identify what the properties of that system is going to be and that is the beauty of particle simulation.

The second thing is we choose a system is we mean both the molecules that I am trying to simulate and the conditions of interest and I have been telling you about the idea of ensembles we can identify what are the control variables in the problem and put the system in that particular control variables to that extent our simulation will have the same control variables as the experimental situations and therefore mimic the experimental situation without having to do the experimental lab.

The third thing that is also very important to know and that relates the computing power that we have is that the system size is not the same as the actual size of system that I want to mimic actually we will be doing very smaller system in comparison to that, not only because it is computationally not possible but also because it is typically not necessary in many cases. Simulations of much smaller systems can still give me the same thermodynamic behavior as the actual system and this is what we have discussed earlier.

(Refer Slide Time: 02:52)



Practical Aspect #1: What to Simulate?

- We do not need to synthesize a system to be able to simulate it!
- By system, we mean both the concerned particles or molecules and the environment (e.g., solvent) under conditions of interest (temperature and pressure)
- Simulated system size is NOT the same as (usually much lesser than) actual system size
- System size should be large enough to contain a "representative" system that mimics actual system, and provide good statistics
- System size should be small enough to be simulated within a reasonable time frame with available resources
- Ideally, one should look a convergence of desired properties with increase in system size, but it is not always possible

So, we have to choose whatever we can say a representative system that can mimic the actual system and provides good statistics to find the thermodynamic behavior but we do not have to actually model the precise system that is the dimensions of for example the beaker in which a solution is located having said that, it should be like the key interest in whatever we are doing is to make sure that we have convergence of the properties that I am trying to get from the simulations until the convergence is not obtained, until the averages we obtain add water reliable enough, we cannot be certain about the results of our simulations.

So, whatever we are doing whatever system we are simulating, we have to mix your that the properties converse averages are reproducible or correct to our whatever error we can we can tolerate.

So, the next thing that we have discussed is we have we can choose the periodic boundary condition in most cases. This is good enough when we try to simulate the bulk behavior of system from simulations performed over smaller system sizes. It avoids artifacts that may possibly arise if you are using some artificial hard boundaries for example, let us say elastic walls and then when we come to what to compute we typically care only about averages we typically do not care the property value at a particular time or at a particular state in a Monte Carlo simulation. We are typically interested in the averages over many, many of those frames and it is those values that really make sense not the value we obtain for a particular state because that state may be realized at a time or may not be realized only when we look at the averages over many, many frames then only we can predict the actual equilibrium behavior so that is itself defined in equilibrium sense.

Now that average we can define multiple ways we can define as a time average, we can define as a spatial average or we can define as an ensemble average. Let us say for example, when we are doing Molecular Dynamics, we are interested in time averages. When we are doing a Monte Carlo we are interested in the ensemble averages but nonetheless both these averages are actually same when the system is ergodic and that is the assumption we typically make in all the particles simulations.

Then when we actually conduct the simulations there are essentially two phases in simulations both in Monte Carlo and Molecular Dynamics first there is an equilibration phase were we make sure that whatever initial state we started with that evolves to an equilibrium state and we define the equilibrium state as that in which the properties begin to fluctuate around some average value because that is the definition of equilibrium it is not that you will have a state of zero fluctuation, but you will start to have both positive and negative fluctuation almost in similar amounts around them average value.

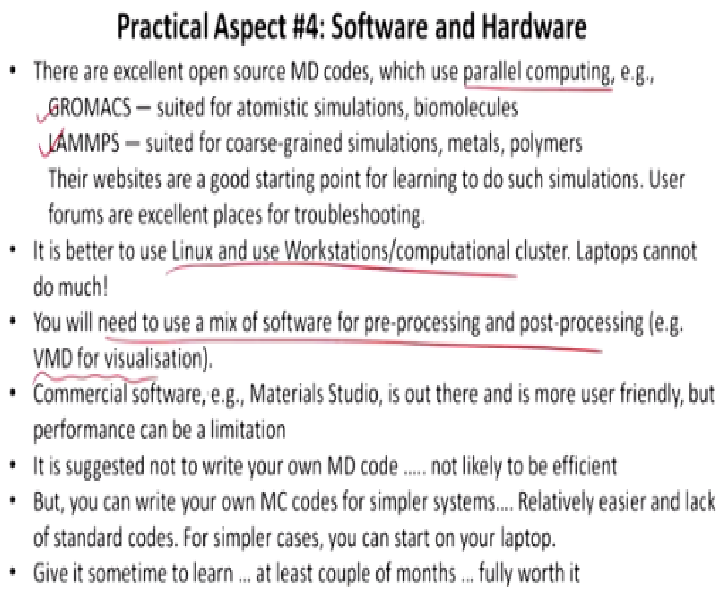
And finally what we have to keep in mind is that equilibration may happen over the times we are simulating or may not happen, so that is where our judgment of if equilibration has occurred or not is sometimes flawed, the reason is let us say if I am simulating for the 100 nanosecond and over that 100 nanosecond the properties appear to have conversed but let us say if I do simulations for 100 times more or if I would have done simulations for 100 times more we are never sure if we will get the same average as that I have obtained for a much smaller time simulations. So, our judgment of the equilibration may sometimes be flawed and therefore the good idea is always to simulate for as much time as possible but then we are limited by the computational power so we have to always make some kind of a compromise in terms of like how we define the equilibrium because we cannot do very, very long simulations.

And therefore we should keep in our mind that if someone repeats the simulation for 100 times more time or 100 times larger system size, whether they will get the same equilibrium or not is not really guaranteed, this is what we hope to have but this is not really guaranteed there is no clear mechanism that can guarantee that.

The last thing is once we have the system equilibrated then we do a production phase. Now each of the frames that will be generated the trajectory that is generating I will average over that basically compute properties of interest in each of these frames actually some of these frames will come to that point later and we average over the properties over those frames that I am collecting and compute the average property, so we really discard the equilibration phase when it comes to the prediction of behavior but equilibrium phase is necessary to ensure that properties be compute are independent of the initial state where I started from really have to come to the same equilibrium state irrespective of where I started from.

The last thing comes down to the software's and hardware's we can use, so I early on I have so do you solve the Monte Carlo codes in fact most of the Monte Carlo simulations you can really code it is much easier to code than compared to Molecular Dynamics. Molecular dynamics itself the algorithm is quite simple but then there are details that we will discuss that makes it somewhat complicated but more importantly we really want to use the power of parallel computing because molecular dynamics is rather easy to parallelize will come to that point again and therefore we should use the codes that really can make use of the high performance computing which is using like some kind of a parallel computing framework and therefore it is very important that we use highly computationally efficient codes and thankfully there are some very good codes that are in open source available which we can use and we do not have to really reinvent the wheel and start writing our own MD codes unless there is a very pressing need in Monte Carlo it makes sense but in Molecular Dynamics it is typically not suggested that we should write our own codes because to generate a code of efficiency similar to GROMACS and LAMMPS are probably one of the difficult thing to do and that is where we should use the best available software's out there.

(Refer Slide Time: 09:42)


Practical Aspect #4: Software and Hardware

- There are excellent open source MD codes, which use parallel computing, e.g.,
  GROMACS – suited for atomistic simulations, biomolecules
  LAMMPS – suited for coarse-grained simulations, metals, polymers
  Their websites are a good starting point for learning to do such simulations. User forums are excellent places for troubleshooting.
- It is better to use Linux and use Workstations/computational cluster. Laptops cannot do much!
- You will need to use a mix of software for pre-processing and post-processing (e.g. VMD for visualisation).
- Commercial software, e.g., Materials Studio, is out there and is more user friendly, but performance can be a limitation
- It is suggested not to write your own MD code ….. not likely to be efficient
- But, you can write your own MC codes for simpler systems…. Relatively easier and lack of standard codes. For simpler cases, you can start on your laptop.
- Give it sometime to learn … at least couple of months … fully worth it

So, GROMACS is pretty good for atomistic simulations particularly biomolecule or protein simulations on the other hand LAMMPS is pretty good for coarse grained simulations, simulations of metals and polymers and all that. And you can pretty much get all the information regarded learning these software's on their website. I will not go into details of any of these software's and their user forums are again excellent places for troubleshooting. So, I will do these are the two open source most common software's, these are not the only ones I will give you a list in a minute, but these are the ones which are I would say mostly used in the molecular simulations.

I strongly suggest if you are coming from windows background to start using Linux because most of the software's really work either in Linux or some kind of like a Virtual Linux installed in a windows machine. It is always better to or be comfortable within the Linux to use the software's, we typically want to run simulations or computer stunt clusters and therefore Linux comes more handy in doing these simulations.

Now it turns out that when we are using GROMACS or LAMMPS much of the pre-processing or post-processing work we have to do with some other tools out there and there are many of them. For example I showed you example of VMD for visualizing the movie of how the particles you are moving, similarly there are other tools to develop for example the force field the system itself for example developed the molecular structures and so on and there is a whole host of them many of them are open source as well which can be used as required.

Similarly there are many post-processing tools once we have the coordinates and I want to print the properties, of course GROMACS and LAMMPS has already many tools to do analysis but in many cases you may have to use some other tools part from their standard tools available in closer the software. There are commercial software's like Materials Studio, they are pretty good they also have very nice GUI interface as opposed to GROMACS and LAMMPS. However when you are using the GUI kind of an interface you may have some limitation with performance. Keep in mind that all these simulations are computationally intensive, so if you are able to see the movie happening or MD simulation going on or the particles moving in a simulation that graphical thing is also adding to your computational burden so I will do they may look nicer it is preferable that we work in I would say a cell like kind of an interface and I do not use the GUI interface way too much especially when using the running the code because that is where we wanted to be highly efficient.

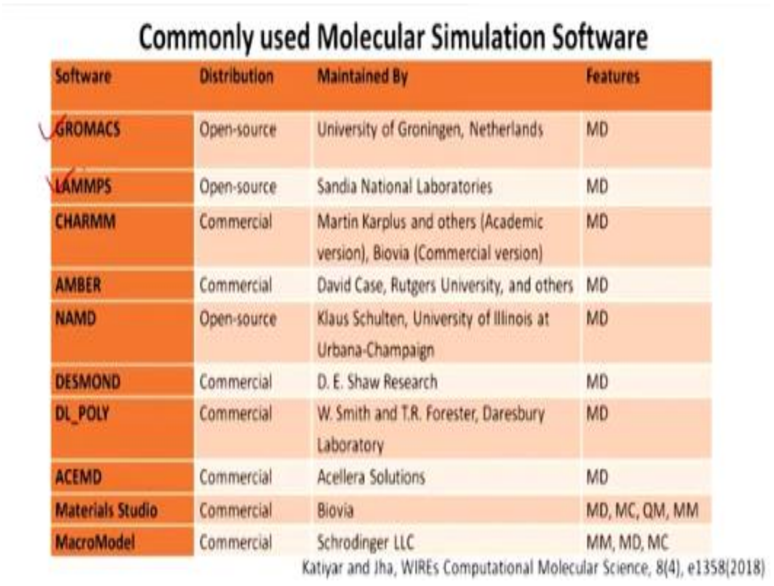
In the analysis part once we have trajectories we can use all the tools out there, but in the core of the simulation it is recommended that we do not use this GUI kind of an interface. Having said that the material studio is a very rich software in terms of many, many tools that are out there and it can also be run in cell mode where the performance is sometimes as good as GROMACS and LAMMPS.

We can also write your own molecular dynamics port, they may not be always efficient if you are not having a strong background in computational programming, so I will again recommend to use software so if possible if it is not then of course you can use it for some kinds of algorithms that you want to develop if that thing is not there you can write your own code many times you can also add that feature in the standard open source software's then writing the everything from scratch because you can still use all the other part of the algorithm from the GROMACS or LAMMPS and you can add your small bit, for example some kind of force field you want to include or some kind of algorithm you want to improve that can pretty much be done in any of these software's or other software's.

Having said that the Monte Carlo we can pretty much. It turns out the Monte Carlo is somewhat difficult to parallelize and most of the applications of Monte Carlo comes in the coarse grain simulations which are relatively simpler systems when you are using the Monte Carlo for atomistic simulations again you use some specialized codes out there but for that cases MD is more commonly used because MD is easier to paralyze.

Having said this I must say that both Monte Carlo and Multiple Dynamics both of them are worth learning use it. Set aside some time apart from this particular course and that time you spend in learning it I must say will be fully worth it is not like a one day thing that I tell you and you learn the software. You have to spend some time learn what are the all the features in this software's and once you learn it, it is quite fulfilling in my opinion.

(Refer Slide Time: 16:02)



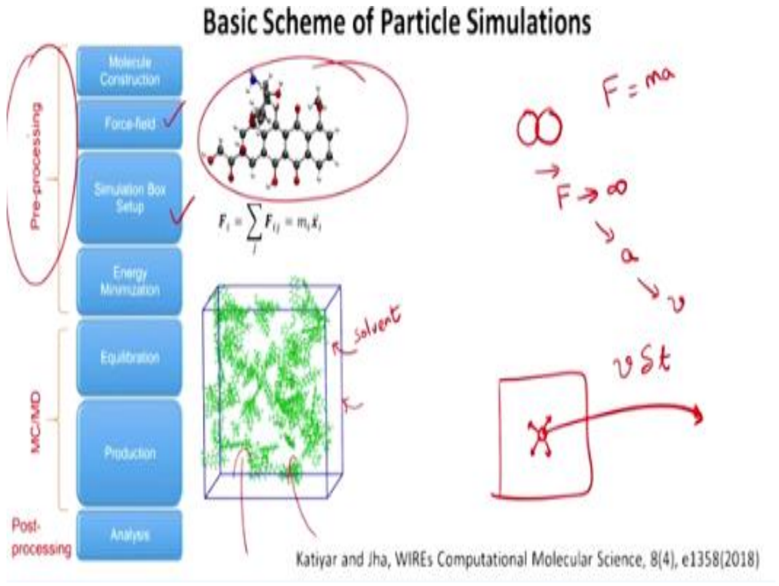
Commonly used Molecular Simulation Software

Software	Distribution	Maintained By	Features
✓GROMACS	Open-source	University of Groningen, Netherlands	MD
✓LAMMPS	Open-source	Sandia National Laboratories	MD
CHARMM	Commercial	Martin Karplus and others (Academic version), Biovia (Commercial version)	MD
AMBER	Commercial	David Case, Rutgers University, and others	MD
NAMD	Open-source	Klaus Schulten, University of Illinois at Urbana-Champaign	MD
DESMOND	Commercial	D. E. Shaw Research	MD
DL_POLY	Commercial	W. Smith and T.R. Forester, Daresbury Laboratory	MD
ACEMD	Commercial	Acellera Solutions	MD
Materials Studio	Commercial	Biovia	MD, MC, QM, MM
MacroModel	Commercial	Schrodinger LLC	MM, MD, MC

Katiyar and Jha, WIREs Computational Molecular Science, 8(4), e1358(2018)

So, these are some softwares apart from GROMACS and LAMMPS these are the first two are open source but there are softwares like CHARMM and AMBER they are also pretty good NAMD is also open source that is also used a lot in protein simulations and all that. Then there are others like DISMOND, DL POLY, ACEMD, Materials studio and Macro Model. I do not have a clear picture of the current status of the performance of all of these software's. In my opinion all these are pretty good it really depends on the need that you have and the licenses that you have of what you can do, but if you do not have the licenses it is better to go with open source persons which are at least as good as any commercial software if not better.

(Refer Slide Time: 16:50)



So, this is the basic scheme of particle simulations be it Monte Carlo or Molecular Dynamics there are there may be some additional steps in some specialized simulations but this pretty much covers I would say most of the standard simulations.

So, we start with constructing the molecules or models that I want to simulate. So, I am giving you an example of a molecule it is a drug molecule and then we choose a force field typically force field is something that depends on the chemistry I am trying to simulate and force fields are typically external to the software that you are using to typically have to borrow the force field from somewhere else software's may provide an interface for doing that but typically force fields are not part of any standard software you may have to think of what force field to use and we can find the parameters of that force field in the web or on software itself.

And then we set up the simulation box that means we start with maybe a cubic box or some other safe box that is what I am doing right here. You fill the box with molecules. If you are working with for example a liquid state then you pretty much can randomly place the molecules in the box as many you want to do. If we are doing in gaseous state the same we can do pretty much sometimes when we are doing a solid state or a soft material state in that case, we need to think of more sophisticated manners.

Let us say I want to simulate a bilayer or an interface then I have really construct some kind of a surface and there are tools out there that can do that but in most cases randomly filling the box with molecules is all that is needed as the initial state. Then in addition to the molecule themselves they may be in some solvent. So, in this case this blue dots are solvent molecules in this case water.

So, you have to fill the box with water molecule to have the density of the system that you want to simulate and after doing that and before doing the MD or MC we typically do an energy minimization step and that is required particularly in atomic systems because sometimes when you are generating the molecules and filling the box with it there are instances where some atoms may be overlapping or they can be very close to each other.

Now if I start the simulation from that state what may happen is that you will get very large forces or energies because the initial configuration was wrong there was an overlap that should not have been there. So, energy minimization basically remove those overlaps and brings the system in rather stable state so that if I start from there the forces and energies are well behaved just to think of like what a very high force can do in your simulation.

Let us say for example, there are two atoms which are somehow overlapping in the initial state because we are randomly generating it they are overlapping and this for example gives you a force value that is very large that is we can close to infinity in a hard sphere model, but can be a very large number even for a soft sphere model or some other model. So, this will give me since I am solving F is equal to ma, a very large acceleration in turn this will give me a very large velocity.

So, what is going to happen in a very small time step it will tend to move v δt and v is large so we δt is also large so an atom in the system may have a movement that is more than the box size and if many atoms start doing that then we have some kind of an explosion happening just because we started from a very poor result configuration, really want the system to relax slowly we do not want attempts to really undergo weight drastic movements because those will make our codes numerically unstable, we will come to the integration part and we can talk about that the errors of integration will also be larger in that case, but more importantly what we really want to make sure be it Monte Carlo or Molecular Dynamics is that the movements of the particles themselves are not really very drastic at least not larger in the box size, it should be much smaller than the box size if it is comparable to that then we should either make the box bigger or we should see if there is some overlap in the initial state if there is something wrong that we have done or can we reduce the time and step or can we reduce the step size in a Monte Carlo simulations.

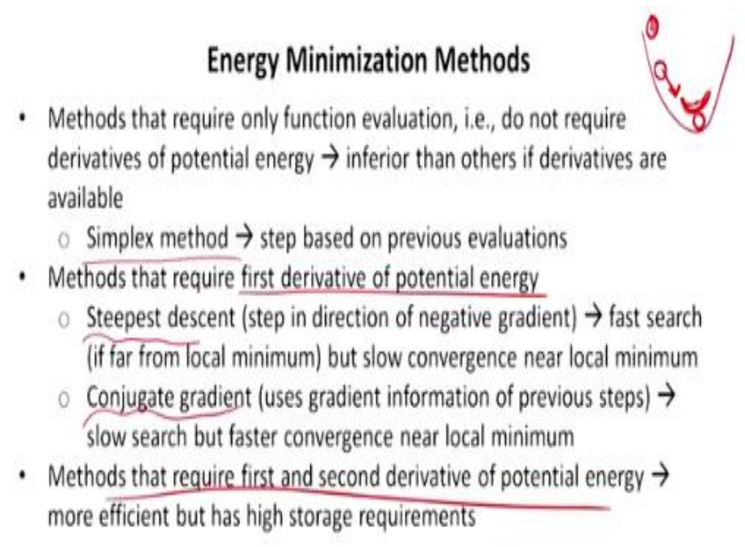
So, all these steps comes under the pre-processing part of the simulation. So, we construct the molecule choose the force field develop the simulation box by first filling in the molecules then filling in solvent and then we minimize the energy and our system is now ready to simulate. Now we can do a Monte Carlo or Molecular Dynamics in both these cases, there are two phases and equilibration phase as I said we look for the convergence of property of interest and then a production phase where we compute the averages and once we are done with that then there is a post-processing part where we can analyze the coordinates and momentum.

So, what we have to we have to keep in mind is when we are doing the Monte Carlo or Molecular Dynamics, typically we are simply storing coordinates and momenta we are not even checking for equilibration. So, the standard practice is first we run for long enough time and then we analyze find properties and see whether we had equilibration or not and if it has not occurred we can start with the last saved coordinates and momenta and continue from that point and run for longer. So, it is better to keep analysis apart from the simulation not only because it will add to the computational burden but also because if I later on think of computing something else I do not have to redo my simulation.

So, we should keep the analysis part always separate from the main coding part we can decide afterwards once the code has run whether we have reached equilibration or not and if it has not run then we can of course run for longer time.

So, before we get to the MD part, let us briefly discuss the energy minimization. So, there are essentially three methods that can do that: the first method is based on the idea of only function evaluation if we do not have the derivatives of energy in this case we go for these methods, there are simplex methods here we make small steps again these methods are not same as the particles from different methods like Monte Carlo and MD these are just done to minimize energy to an extent that the system is in a good initial state and therefore these are I would say separate from the actual simulation that we are doing.

(Refer Slide Time: 24:27)



Energy Minimization Methods

- Methods that require only function evaluation, i.e., do not require derivatives of potential energy → inferior than others if derivatives are available
  - Simplex method → step based on previous evaluations
- Methods that require first derivative of potential energy
  - Steepest descent (step in direction of negative gradient) → fast search (if far from local minimum) but slow convergence near local minimum
  - Conjugate gradient (uses gradient information of previous steps) → slow search but faster convergence near local minimum
- Methods that require first and second derivative of potential energy → more efficient but has high storage requirements

So, we can use the simplex method where we make steps that are based on the previous evaluations. Let us say for example, I am doing minimization over 50,000 steps then every step decides the previous step of minimization and every time I compute the function value or the energy value. If we have the derivatives of the energy available to us that is typically the case in most MD systems in that case we can use methods such as steepest descent or conjugate gradient that essentially work on the idea of following the gradient.

So, let us say for example, if energy of the system is something like this and I start from here then the system will tend to go downhill because we follow the gradient that is the idea of steepest descent, conjugate gradient is slightly more slightly smarter in this case, we also use the gradient data from the previous step where the system has been seriously. It turns out the steepest descent is pretty good in most cases it gives rise to a faster search if we are far from the minima but it has rather slow convergence near the minima point this may have certain fluctuations around the minima.

In the case of conjugate gradients, the search itself may be slower if we are far from the minima but the fluctuations are lesser than in this case we typically have faster convergence near the minima even better our methods the very we can use the first and second derivative of the energy these are clearly more efficient but you may imagine that I now have to store the second derivative of the energy and that will of course require more storage and therefore in most software's the typically use either the first or the second method, second one is the most common one particularly the steepest descent and conjugate gradient methods are commonly used in both minimization.

I want to make a point here that energy minimization itself has some value and it is also used in certain types of analysis where simply energy minimization is being conducted. Let us say for example if I want to know the optimum geometry of a molecule, I can minimize its energy and it can give me the optimum energy. In this particular context that I am discussing in this course, we are using energy minimization only in the preprocessing part.

It is not really a very central part of the algorithm it is necessary only when there are some overlaps as a good practice we may want to do it, but we are not really focused on energy minimization itself we are assuming that our geometries are already optimum there may be some small errors because of the way we are building the system, but if system is really far from the energy minima then in that case all these methods needs more serious thought than what we are doing in a typical MD simulation.

So, let us now come to the heart of the matter that the Molecular Dynamics so, let us start with the equation of motion and Cartesian coordinates. We have earlier discussed that we can work with generalized coordinates, but let us start simple with Cartesian coordinates in any case the simulation boxes that I have been discussing we are mostly cubic so x, y, z coordinate really is the most intuitive choice in those cases.

So, I want to solve a second order ordinary differential equation, that is a derivative of r_i the second derivative of positions.

$$m_i \ddot{r_i} = f_i$$

So, double dot represents second derivatives of r_i with respect to t_i and both r_i and f_i are vectors this is the position of particle i and this is the force acting on particle i.

So, clearly since we are solving a second order differential equation we require two initial conditions typically the initial position and the initial velocity or the first derivative of the position.

$$\frac{dr_i}{dt}$$

We can also write this equation as two first order ordinary differential equation, that is just a compact way of writing it that makes it somewhat easier to work with. So, we can say-

$$\dot{r}_i = \frac{p_i}{m}$$

we are p_i is the momenta defined as mass multiplied with the velocity

$$p_i = m_i v_i$$

We can have m_i here if the masses are identical for all the atoms in the system or all the atoms are identical in that case we can use simply m otherwise we can use m_i.

$$p_i = f_i$$

So, if I divide the momenta by the mass we clearly get the velocities and then we can write p_i dot, as f_i that is simply adding statement of the second Newton's law of motion. And then we will have the initial condition same as earlier the initial position and the initial momentum.

So, this is the statement of the MD problem in the next lecture we will see how do we numerically solve it and in addition to that we also discuss how do we control the temperature and phase pressure in an MD simulation that is required if I want to do simulation in different ensembles as opposed to the micro canonical or NVE ensemble in which these equations are valid.

So, in this case by default, the MD algorithm is using the NVE ensemble in Monte Carlo by default the algorithm is for the NVT ensemble. So, now if I want to do a temperature control or a pressure control what I want to work in a canonical ensemble or isothermal isobaric ensemble we need to have a mechanism for temperature and pressure control and this is what I will also discuss the next lecture.

So, with that I want to conclude thank you.