Computer Lab 3b - Molecular Dynamics Simulations

The protocol for running the MD program is very similar to the one we used for the MC program. We will look at a few differences in the code and then start our MD program.

I. **Anatomy of a Molecular Dynamics simulation program**. Obtain the files to be used in this lab, put them in your personal directories on the cluster and look at the contents of the main program file *md.py*:

1. Login to the Mac, change to your permanent home directory, launch tcsh, make a subdirectory called *lab3b* and change to that subdirectory.
2. Open a second terminal window on the local machine and **`ssh`** to your account on the cluster (**`compbio2@kirin.kit.jhu.edu`**), cd to your JHEDID subdirectory and initialize the tcsh UNIX shell, make a *lab3b* directory in your cluster directory and cd to that new directory.
5. On the cluster get the tar archive of files for this lab, unpack it and cd to the new directory.

```
cp /home/compbio2/Shared/md.tar .
tar xvf md.tar
cd md
cp /home/compbio2/Shared/normal-python3 .
```

II. **Run MD simulation on Argon box.** To save you time the instructor has done an equilibration MD run and prepared a restart file for you. So you can start directly with the production run. Email answers to the yellow questions to **achin14@jhu.edu**.

1. During an MD simulation we keep track of both the *positions* and *velocities* of each particle. Therefore, in order to restart the simulation from where it stopped previously we require a file with both types of information. The restart file created for you is called *restart.cv* to differentiate it from the *restart.pdb* created in the MC simulations. On the cluster use the command **`less`** (or **`view`**) to view *restart.cv*. 1. What are the labels for the six columns starting with row 5? If you missed the discussion in class you can look in the **storemd()** function in *modules.py* to see the format of the *restart.cv* file. If that doesn't clear it up look in *md.py* to see where **storemd()** is called and where the passed values come from.

1. 2. Use the *normal-python3* job file to run the simulation starting with the last previous configuration (the *restart.cv* file). Change the following line so that the word `yourJHEDID` is really your actual JHED_ID. Leave the `#$ -N` as is.

   ```
   #$ -N yourJHEDID
   ```

   And edit the command to be executed,

   ```
   $PYTHONBIN md.py md_params_prod restart.cv
   ```


3. Submit the job to the queue,

```
./normal-python3
```

4. Check that your job is running,

```
qstat –u \*
```

5. At the end of your simulation check the the *yourJHEID.o****** file using the **more** command.

2. What is the difference between the two energies given? (Keywords, not a number.)

III. View progress of simulation. We will use PyMOL to view the consecutive PDB files in the large file called *trajMD.pdb*.

1. Use sftp to fetch the *trajMD.pdb* file back to the Mac and look at the contents of the file. Use the command **less** (or **view**) to view *trajMD.pdb*. This file contains multiple configurations of the Argon system in PDB format.

3. What are the keywords at the beginning and end of each configuration?

2. To do a sanity check that our trajectory output is what we expect we will determine the number of configurations in the *trajMD.pdb* file. First find a line that is present only once in each configuration (we will use the line starting with "**MODEL**" which designates the beginning of each configuration). Quit viewing the *trajMD.pdb* file and on the terminal command line enter the following,

```
grep 'MODEL' trajMD.pdb
```

To count how many lines with "MODEL", pipe the output of the `grep` command into another UNIX command `wc` (word count) and tell it to count only lines (**-l**) (the lower case letter L, not a numeral one)

```
grep 'MODEL' trajMD.pdb | wc -l
```

Use the man command to find out how the **–l** argument changes the output of the **wc** command.

4. How many configurations are in the trajectory?

5. What combination of parameters in the *md_params* file determines the number of configurations?

6. Did you get the correct number of configurations?

3. Now we view the configurations. Launch PyMOL, show spheres and color by rainbow

```
pymol trajMD.pdb
```

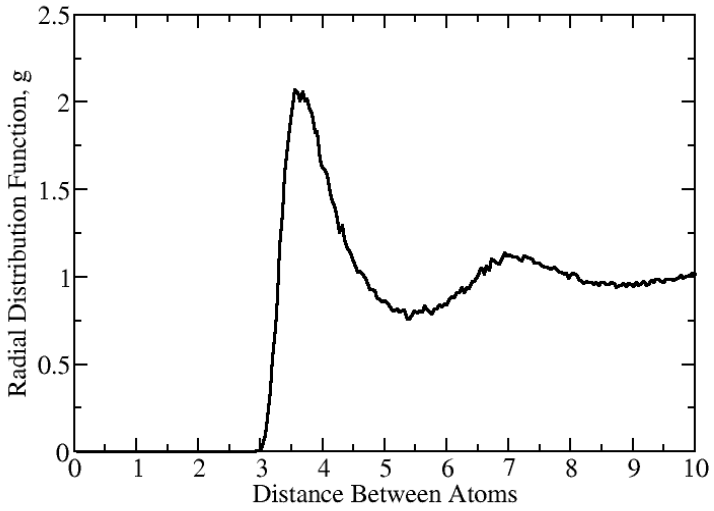

trajMD | S | spheres
trajMD | C | spectrum | rainbow

Are the colors well mixed? In other words, does it look like the system is at equilibrium from a visual inspection?

IV. **Analyze distributions of configurations**. *The immediate goal of molecular simulations is to create an ensemble of Boltzmann energy distributed configurations or conformations*. We want to compare the distributions created by MC and MD simulations to determine if they are the same and if they follow a Boltzmann distribution (which we will approximate as a normal distribution).

The most probable distribution of the Ar atoms we have simulated is when they are separated by a distance representing the most favorable energy. The energy function used in the simulations is the Lennard Jones potential function. Therefore, the most energetically favorable distance between atoms is the van der Waals distance and is equal to $\sigma*2^{1/6}$ or about 1.12σ. A fluid state such as the ensemble of particles we are simulating in lab has no permanent structure. However, there are structural correlations that we may use to provide information about the *average* structure of the system. The **radial distribution function**, $g(\mathbf{r}_1, \mathbf{r}_2)$ is used to describe the average relative positions of particles in an ensemble. It is also called the **density correlation function** or **pair correlation function**.

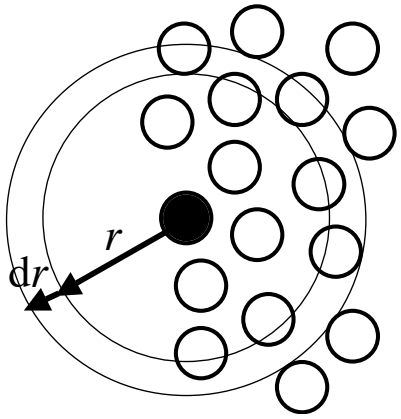
For each atom we calculate the distance to all other atoms and plot the probability of finding another atom at each distance *versus* the distance. The figure below shows a typical $g(\mathbf{r}_1, \mathbf{r}_2)$ plot for a collection of hard spheres of diameter R as a function of distance d between the spheres.



The $g(\mathbf{r}_1, \mathbf{r}_2)$ can be described as the effective local density of particles at position $\mathbf{r}_2$ given particle i at position $\mathbf{r}_1$ *relative to* the mean bulk density of particles (which is 1 in the RDF).

For an isotropic system of identical particles (such as our Ar simulations) only relative separation is important and we can substitute the distance between particles (r) for the individual positions (**r**). **To restate, $g(r)$ can be described as the local density of particles at distance r relative to the mean density of particles at all distances.**

The above definition of $g(r)$ implies that the probability of finding a particle in the volume dr at a distance r from a given particle is $\rho g(r)$dr where ρ is the number density. For these calculations distance is measured between the centers of the molecules. The figure below depicts the case in two dimensions.



To calculate $g(r)$ for a system of isotropic single atom molecules we take every molecule i and calculate the distance to all other molecules j…N and keep track of how many times we find j at a distance interval r + dr. This pairwise summation may be expressed as

$$g(r) = [1/N\ (\sum_i \sum_{j \neq i} \rho_{shell}(dr))] / \rho_{system}$$

where ρ = number density and $\rho_{shell}(dr)$ = the density of molecules in volume shell dr. The radial distribution function is normalized to the density of the bulk system (ρ_{system}) so at large distances between molecules the $g(r)$ approaches a value of 1.0.

The $g(r)$ is an important descriptive parameter for molecular systems and it can be used to calculate other parameters of the system. This is true because **the $g(r)$ is determined by the Boltzmann factors** of the system as seen in the next equation,

$$g(r) = \frac{N(N-1)\int\left(e^{-E(r_1 \ldots r_N)} dr_1 \ldots r_N\right)}{\rho^2 Z}$$

In fact, the potential energy of the system may be calculated from the g(r) using a transform of the above equation.

We will use the g(r) in a qualitative way to compare the MD and MC systems as described below.

A module called *rdf_calc.py* is provided in the tar archive of the simulations which will read in all the configurations in the trajectory (or ensemble) file and output the radial distribution function as an *x*,*y* list suitable for plotting. To run the python module do the following on the Mac after you fetch the files back. In the subdirectory that has both files:

```
python3 rdf_calc.py trajMD.pdb
```

The output file is called *rdf.dat* and you may plot this data using **`xmgrace`** (or your favorite plotting app).

7. What is the average distance between an atom and other atoms in its first neighbor shell?
8. Is this distance the same as the van der Waals distance?

Go to your *lab3a/mc/* directory on the Mac and run the *rdf_calc.py* program on the *ensemMC.pdb* file for MC, load both *rdf.dat* files into xmgrace and compare the RDF plots for both MC and MD simulations.

9. Do both MC and MD give the same distribution of pairwise distances? In other words, do both algorithms give the same *average structure* for the system?

The two plots may not be identical because of random noise. But if you increase the number of steps in each simulation the two plots should become more similar. You should convince yourself that both MC and MD generate populations that have equivalent average structures.

Does this result mean that both distributions will have the same energy?

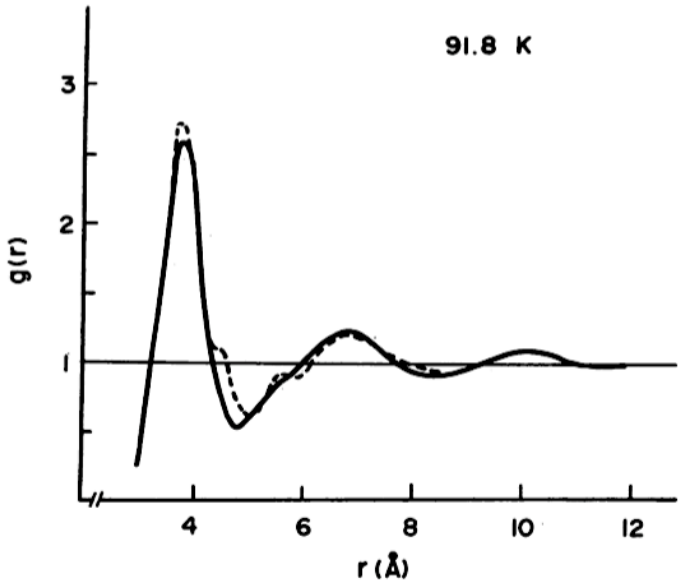
Test this hypothesis by plotting the energies of your production ensembles (MC and MD) as histograms and fit the histograms with the Gaussian formula, **`y = a0 * exp(-((x - a1) / 2 * a2)^2)`**, as you did in Lab1c. You should have a new *epd.dat* file created during the MD run. Fetch this from the cluster and compare the energies in this file to the energies in the *epd.dat* file created during the MC run. Make two plots and compare them. (Use awk to extract the energies from *epd.dat* to a new file; edit that file to remove the column labels. Use **`-130, -100, 20`** for the Start at:, Stop at: and # of bins values for histogram binning and a0 = **`100`**, a1 = **`-100`**, a2 = **`10`** for the initial guesses in the Gaussian fitting expression).

10. Do both distributions of energies have the same Gaussian fit values?

You should leave this lab convinced that both MD and MC generate equivalent Boltzmann distributed ensembles of molecules. This is the first goal of any molecular simulation.

Another take-home message is that correlation functions may be used to *model the structure* of a fluctuating system. Correlation functions of a system of molecules can be obtained from experiment as well as simulation. **They are a link that allows one to**

compare the results of simulations with reality. Below is an example of the RDF of argon at 91.8 K obtained by X-ray scattering experiments (dashed line) compared to the RDF obtained from a simulation such as yours.



From Yoon et al., PNAS 78(11): 6588-6591 (1981)

FIG. 2. Radial distribution functions of liquid argon at 84.4 K (*Upper*) and 91.8 K (*Lower*) in comparison with x-ray measurement. ——, Calculated; ----, x-ray from ref. 9.

Going deeper. Molecular dynamics was first used to simulate the properties of monatomic liquids. The following are three classic papers from the beginning of this field. You are not required to read these (they are heavy going) but if you have an interest in going further they are a good starting point. The paper by Rahman is of special historic interest as the first MD paper. All three have PDF links from the References page of the course web site.

- Rahman, A. Correlations in the Motion of Atoms in Liquid Argon. Phys. Rev. 136(2A):405-411 (1964).
- Street, W.B. and Staveley, A.K. Experimental Study of the Equation of State of Liquid Argon. J. Chem. Phys. 50:2302-2307, 1969.
- Verlet, L. Computer Experiments on Classical Fluids I. Thermodynamical Properties of Lennard-Jones Molecules. Physical Review 159:98-103, 1967.

