Chemistry (4|6)0650 - Computational Spring 2017
Professor J. Daniel Gezelter Due Fri. 3/10/2017

Computational Exercise 3

In this example, we'll run and analyze a molecular dynamics simulation of a simple liquid (methanol) starting with a structure file (in XYZ format). Getting your molecule of choice into a simulation program is never a black box procedure, and will almost always require some hand adjustment of the input files.

Setting up a liquid simulation

1. Use Avogadro, build the methanol structure (CH_3OH), set up the MMFF94 force field, and optimize the structure. Save the structure as an XYZ file called: `methanol.xyz`.

2. Open a terminal, and copy your methanol structure over to your home directory on the CRC machines:

   ```
   scp methanol.xyz user@newcell.crc.nd.edu:~/
   ```

   replace *`user`* with your own Notre Dame NetID.

3. Log in to the CRC machines and load the requisite modules. Some of these steps below will take some time, so if you have access to another CRC front-end node, you might want to use that instead of newcell:

   ```
   slogin user@newcell.crc.nd.edu -Y
   module load openmd
   module load xmgr
   module load vmd
   mkdir chem650
   mv methanol.xyz chem650
   cd chem650
   ```

4. Use the `atom2omd` program to convert the structure into a format that can be read by OpenMD:

   ```
   atom2omd -ixyz methanol.xyz
   ```

   This command will create an incomplete OpenMD file called `methanol.omd` that must be edited before it can be used.

5. OpenMD can use a number of force fields, but in this example, we'll use the Amber force field. If you are using this force field and are starting from an XYZ file or non-standard PDB file, you ***must** edit the atom types*. In the `methanol.omd` file:

   a. change the atom typing for the methyl carbon from `C3` to `CT`
   b. change the `O3` to `OH`
   c. All of the hydrogens on the carbon should be changed from `HC` to `H1`
   d. The hydroxyl hydrogen (`HO`) should be changed to `HO`.

6. At this point it is also a good idea to change the name of the molecule from `MolName0` to something descriptive (use "`methanol`"). This should be done in two places; once in the molecule description and another time in the component block.

7. Before the simulation can run, add a `forceField` line after the `component` block:

   ```
   forceField = "Amber";
   ```

   At this stage, you should be able to run OpenMD on the file to check to make sure your hand-crafted atom typing can be matched up with types known by the force field:

   ```
   openmd methanol.omd
   ```

   If there are any problems, correct any unknown atom types, and repeat until you get an error about the "Integrator Factory".

8. Next, we'll build a lattice of methanol molecules using this initial structure as a starting point. The density of liquid methanol is roughly 0.7918 g cm^{-3}, so we'll build a simple box of methanol molecules using the command:

   ```
   simpleBuilder -o liquid.omd --density=0.7918 --nx=3 --ny=3 --nz=3 methanol.omd
   ```

   This command creates a new system, `liquid.omd` which contains 108 copies of the methanol molecule arranged in a simple FCC lattice. FCC has 4 molecules in the unit cell, so the total number of molecules = 4 * 3 * 3 * 3 = 108. The molecules are packed at a distance commensurate with their liquid state density.

9. To visualize what the system looks like at this stage, you can run:

   ```
   Dump2XYZ -b –i liquid.omd
   ```

   to create a file called `liquid.xyz`. This file can be viewed in vmd, jmol, or any other chemical structure viewer. The -b flag tells this program to convert all of the simulated atom types (CT, OH, H1, HO) into the *base* types (C, O, H, H)

10. Add the following lines below the `forceField` line of the `liquid.omd` file. Be careful to include all of the semicolons, as they terminate each line:

    ```
    ensemble = NVT;
    electrostaticSummationMethod = "shifted_force";
    ```

```
electrostaticScreeningMethod = "damped";
cutoffRadius = 9;
dampingAlpha = 0.2;
targetTemp = 300;
tauThermostat = 1000;
dt = 1.0;
runTime = 1e3;
tempSet = "false";
sampleTime = 100;
statusTime = 10;
```

11. Initial configurations that are created from bare structures typically have no velocity information. To give an initial kick to the atoms (i.e. to sample the velocities from a Maxwell-Boltzmann distribution), you can use the following command:

    ```
    thermalizer -o warm.omd -t 300 liquid.omd
    ```

    This creates a new OpenMD file called `warm.omd` which has initial velocity information.

12. At this stage, a simple simulation can be run:

    ```
    openmd warm.omd
    ```

13. This should complete relatively quickly, and should create three new files:

    - `warm.stat` (reports the status of the simulation as it progresses)
    - `warm.dump` (contains the actual trajectory data)
    - `warm.eor` (an "end-of-run" file that contains the single most recent stored configuration)
    - `warm.report` (an end-of-run statistical analysis of the trajectory)

14. To view the contents of the trajectory file, you'll need to convert the dump file into something another program can visualize:

    ```
    Dump2XYZ -b –i warm.dump
    ```

    will create a new file `warm.xyz`. To visualize the trajectory, you can use:

    ```
    vmd warm.xyz
    ```

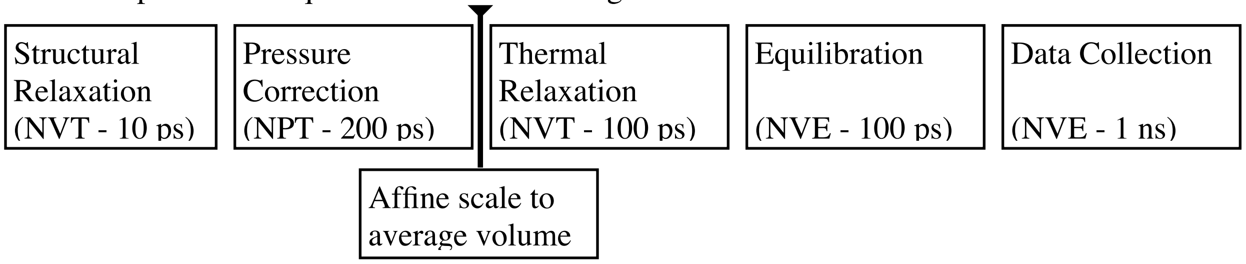
15. The "End-of-Run" file, `warm.eor` can be re-purposed as the starting point for a new simulation,

```
cp warm.eor stable.omd
```

Edit the `stable.omd` file, and change parameters you'd like to change before running openmd on the new file.

Thermal relaxation, Equilibration, Data Collection

MD simulations should go through some very important stages before they are used for sampling and data collection. Hand-made initial configurations are often far from equilibrium structures and will need a substantial period of equilibration. If a simulation starts in a strained or overly dense structure, there may be many stages required to equilibrate that structure. For example, we will often perform a sequence like the following:


Structural Relaxation (NVT - 10 ps)
Pressure Correction (NPT - 200 ps)
Thermal Relaxation (NVT - 100 ps)
Equilibration (NVE - 100 ps)
Data Collection (NVE - 1 ns)
Affine scale to average volume

This gives the structure a chance to relax and thermalize (under constant volume conditions) before the box is allowed to change shape (under constant pressure conditions). After a reasonable volume has been found for a particular pressure, the system is affine scaled to the new box geometry, and allowed to relax thermally (under constant volume). It may be reasonable to collect *structural* or *configurational* data in isobaric-isothermal (NPT) or canonical ensemble (NVT) simulations. However, all data collection for *time dependent quantities* ***must*** *be done in the microcanonical (NVE) ensemble!* It is impossible to stress this point enough. All methods for sampling constant pressure or temperature introduce dynamical perturbations. If you care about dynamical quantities, the *only* useful ensemble is the microcanonical ensemble.

In what follows, we will perform a (short) thermal relaxation, followed by equilibration and data collection. At the end of these stages, we will analyze the trajectory to obtain both structural and dynamical quantities for our liquid. Some of these stages will take a while, so if you have access to other CRC front-end nodes, you might want to use those instead.

1. Edit your `stable.omd` file, and modify the following lines:

   ```
   tauThermostat = 100;
   runTime = 1.5e4;
   sampleTime = 1000;
   ```

   run the command: `openmd stable.omd`

2. When this simulation has completed, you can view the various quantities that were tracked during the simulation:

   ```
   xmgrace -nxy stable.stat
   ```

zoom in on the blue line (temperature) and observe that the temperature oscillates around the target value (300) with a period controlled by the `tauThermostat` variable, but eventually damps out to the target value. The other lines correspond to other status variables:

- black: total energy (kcal mol^{-1})
- red: potential energy (kcal mol^{-1})
- green: kinetic energy (kcal mol^{-1})
- blue: temperature (K)
- yellow: pressure (atm)
- brown: volume ($Å^3$)
- grey: conserved quantity (usually kcal mol^{-1})

3. Copy the endpoint of the stabilization run into the starting point for an equilibration run:

```
cp stable.eor equil.omd
```

4. Edit equil.omd, and change the following lines:

```
ensemble = NVE;
runTime = 2.5e4;
```

   run the command `openmd equil.omd`

5. When this simulation has completed, you can check the energy conservation using `xmgrace -nxy equil.stat` Make sure that the total energy (black) and conserved quantity (grey) lines have no drift, and are reasonably conserved during the simulation. There should also be no obvious ringing in the temperature (blue) or pressure (yellow) traces during the simulation.

6. Copy the endpoint of the equilibration run into the starting point for a collection run:

```
cp equil.eor collection.omd
```

7. Edit `collection.omd`, and change the following lines:

```
runTime = 2.5e5;
sampleTime = 500;
```

   run the command `openmd collection.omd`

Analyzing the results: Structural features

1. First analyze the trajectory (dump) file to make a carbon-carbon pair correlation function, $g_{CC}(r)$:

```
StaticProps -i collection.dump -g --sele1="select CT"
             --sele2="select CT"
```

this command should be all on one line. View the $g_{CC}(r)$ function:

```
xmgrace collection.gofr
```

2. `sele1` and `sele2` are selection scripts that can be used to specify other pairs of objects in the system. What does a $g_{OO}(r)$ look like? What about $g_{OH}(r)$? Remember that there are two types of Hydrogen you can query.

Analyzing the results: Dynamic properties

1. First, compute the mean squared displacement as a function of time, $R^2(t) = \langle|\vec{r}(t) - \vec{r}(0)|^2\rangle$

   ```
   DynamicProps -i collection.dump -r --sele1="select methanol"
   ```

   View the *$R^2(t)$* function:

   ```
   xmgrace collection.rcorr
   ```

2. The Einstein expression for the diffusion constant is

   $$D = \lim_{t\to\infty} \frac{1}{6t}\langle|\vec{r}(t) - \vec{r}(0)|^2\rangle$$

   Find the diffusion constant for methanol by finding the best-fitting slope of the *$R^2(t)$* function and dividing by 6. Compare this to experimental values.

3. Compute the velocity autocorrelation function, $\langle\vec{v}(t)\cdot\vec{v}(0)\rangle$,

   ```
   DynamicProps -i collection.dump -v --sele1="select all"
   ```

   confirm that the data you have collected results in an undersampled (rough) velocity autocorrelation function because the configurational sampling was too infrequent.

4. To refine the sampling time scale, copy `collection.eor` to `fine.omd`, and edit the following lines:

   ```
   runTime = 1e4;
   sampleTime = 10;
   ```

   re-run `openmd` on the `fine.omd` file, and recompute the velocity autocorrelation function. If your correlation functions are still rough, re-do the sampling at shorter time scales until you get a smooth autocorrelation function.

Homework

1. Submit publication-ready figures of the following quantities:

    a. pair correlation functions for all of the pairs of atom types present in the system.

    b. mean-squared displacement.

    c. velocity autocorrelation function.

2. Use Dump2XYZ, vmd, and your data collection trajectory to make a movie of liquid methanol that you could show in a group meeting. Put this up on the web and email the URL to Dr. Gezelter. You might find the -m flag for Dump2XYZ useful in keeping all of the molecules in the central simulation box. In VMD, you can easily make movies using: `Extensions → Visualization → Movie Maker → Movie Settings → Trajectory` (rather than rock-roll)

3. Estimate the diffusion constant using the Green-Kubo relationship,

$$D = \frac{1}{3}\int_0^\infty \langle \vec{v}(t)\cdot\vec{v}(0)\rangle dt$$

and your computed velocity autocorrelation function. How does this compare to your estimate from the Einstein relation?

4. Use the `vcorr2spectrum` program to compute the power spectrum of the velocity auto-correlation function,

$$\rho(\omega) = \int_{-\infty}^{\infty} \langle \vec{v}(t)\cdot\vec{v}(0)\rangle e^{i\omega t} dt$$

(Note that you could also do this by symmetrizing your velocity autocorrelation function around *t=0* and performing a discrete Fourier transform. `xmgrace` is a powerful tool for doing tasks like this.)

Plot the power spectrum, find the frequencies of any peaks you see, and assign these peaks to particular features of the molecule.