Calculating Properties of Bulk Water with Molecular Dynamics

Research Questions

Water is a common solvent and plays an important role in many chemical reactions. It has been widely studied and is the topic of this laboratory.

Many properties of water--including several of the anomalous ones--can be explained by its structure, the way water molecules are arranged relative to one another. If you were investigating solid water, you could report the regular positions and repeating distances between molecules. Liquids do not have these features, and instead the radial distribution function gives insight into how ordered the liquid structure is, in the local vicinity of any given molecule. **What is the structure of liquid water?**

The rate of some chemical reactions, including many catalyzed by enzymes, are limited by the actual motion of the reacting molecules as they travel through all other molecules that make up the solution. These reactions are classified as “diffusion controlled,” where their motion is characterized by the diffusion coefficient. It seems like the ability of molecules to move through a solution should change as the temperature of the liquid changes. **How do the dynamics of liquid water change with temperature?**

In order to answer these questions, you'll be simulating the positions and motion of water molecules by modeling their individual structures and the interactions between molecules. Depending on which assumptions you make about the individual structure and molecular interactions, you might get different results for the radial distribution functions and diffusion coefficient. Luckily, there's experimental data for both of these quantities at room temperature. **How well does your model of liquid water and set of assumptions match experimental results for dynamics and structure?** Once you know this, you'll be able to answer the other two questions with confidence that the model works (or with caveats that it doesn’t!)

Background Information

Liquid Structure: Radial distribution function

The two extreme scenarios of molecular interaction are solids, having molecules with strong interactions, which are aligned in a rigid crystal, and gases, having molecules with none to minimal interactions, which are not arranged in any structured way. The amount of interaction (and thus structure) of molecules in a liquid falls in the middle. The structure in a liquid can be characterized by a radial distribution function, which is abbreviated RDF and is the name of the function g(r). This function *g(r)* describes the average number of water molecules in the volume of a sphere with radius *r* from a given molecule.[1]

RDFs can also be constructed for specific atom pairs (*e.g.* how many oxygen atoms are in a spherical volume with a radius *r* from a given oxygen atom). Three RDFs of water determined from a neutron diffraction experiment are shown in Figure 1.[2]
RDFs are related to the probability of finding molecules at specific distances from the given molecule. For a monatomic gas, *g(r)* is flat, because any atom is a random distance from all other atoms. There is no preferred distance; all are equally likely. For a

crystalline solid, *g(r)* has regular, sharp peaks, because atoms are regularly spaced from each other. Only some distances are likely; others are essentially forbidden. Liquids are structured locally but random at a distance, and hence have a few peaks – which correspond to solvation shells, as in Figure 2. Eventually, as the nonlocal density becomes identical to the bulk density, *g(r)* flattens, as seen in Figure 1.

For water, the amount of local structure can be characterized by the first peak, which represents the first solvation shell. The position of the peak indicates how far away from a given water's oxygen the oxygen atoms of nearby water molecules will be, and the height of that peak is indicative of how many nearby water molecules there are (to get an actual number of molecules in the first shell, you'd need to integrate $4\pi r^2 g(r)$ over the range of the first peak).

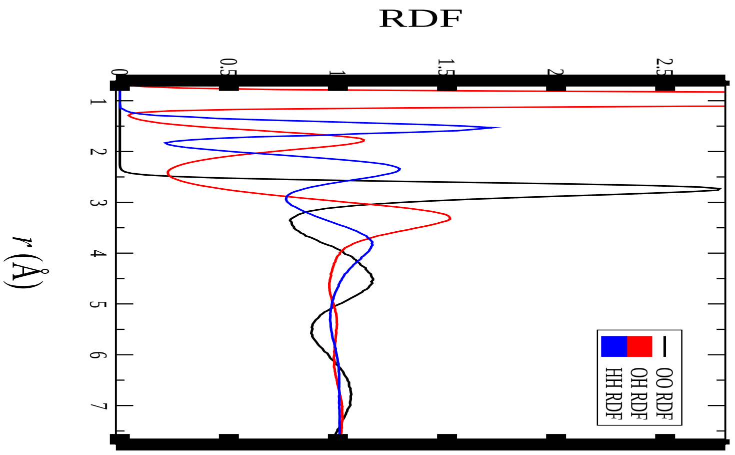


Figure 1. Experimental radial distribution functions of water. The oxygen-oxygen (OO) RDF in black shows the distribution of the first solvation shell, between 2.5 and 3.3 Å. The second peak corresponds to the second shell and its magnitude is determined by the tetrahedral character of the liquid. The oxygen-hydrogen (OH) RDF in red has a large peak at 1 Å due to hydrogens bonded to the oxygen. The region of the second peak corresponds to h-bonded hydrogens, and the third to non h-bonded hydrogens. The hydrogen-hydrogen (HH) RDF in blue has a large peak at 1.5 Å due to hydrogens in the same water molecule. The HH RDF also contains structural information, but in a less clear way than the other RDFs.[2]

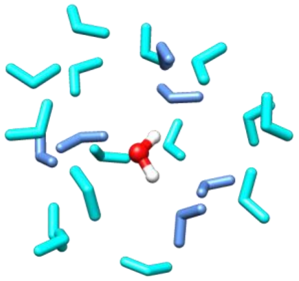


Figure 2. An example of the first two solvation shells in liquid water. The dark blue indicates the first shell, where the oxygens are all roughly the same distance from the red oxygen. The hydrogens, however, might be closer or farther depending on the orientation of the blue molecules, which accounts for the more complicated OH and HH RDFs. The light blue molecules are the second solvation shell; some appear closer only because of perspective.

Dynamics: Diffusion

Thermal fluctuations in water cause each molecule to wander from its original position, executing a random walk. That is, some motions of the molecule are away from its starting point, and some motions are back towards it. Eventually, however, at a rate characterized by *D*, the diffusion coefficient (measured in area per time, often using units like Å^2/ps or cm^2/s), each water molecule will be far from where it started.

The diffusion coefficient is a proportionality constant which relates the amount of particles moving through a given volume to the concentration gradient. Effectively, particles can move faster through a less dense liquid and in the direction where the liquid is becoming less dense.

Mills determined the diffusion coefficient in water to be 0.2299 ± 0.0004 Å^2/ps at 300 K.[3] In terms of other solvents, water's diffusion is similar to toluene (0.2267 Å^2/ps) and higher than cyclohexane (0.14170 Å^2/ps).[4] Water's diffusion is typical of small molecules and is one of its few non-anomalous properties.[5]

The most common way to calculate *D* is to use Einstein's relation, valid for long time scales, that converts a messy integral equation into one dealing with position vectors:

$$\boldsymbol{D} = \frac{\mathbf{1}}{\mathbf{6}}\frac{\langle|\vec{\boldsymbol{r}}_{\boldsymbol{i}}(\boldsymbol{t}) - \vec{\boldsymbol{r}}_{\boldsymbol{i}}(\mathbf{0})|^{\mathbf{2}}\rangle}{\boldsymbol{t}} \tag{1}$$

where $\vec{\boldsymbol{r}}_i$ is the position of the *i*th particle at time *t*.[6] The quantity $\vec{\boldsymbol{r}}_i(t) - \vec{\boldsymbol{r}}_i(0)$ is the difference between the position of molecule *i* at some later time *t* and that molecule's original position at time zero. The angle brackets < and > indicate an average over all of the positions of the molecules in the simulation at time *t*.

The average quantity $\langle\vec{r}_i(t) - \vec{r}_i(0)\rangle$ can be calculated for each time step, so then ***D*** **becomes 1/6 the slope of this average versus time**. For water, the movement of the molecule is associated simply with the oxygen's movement, as the center of mass does not differ significantly from the oxygen's position (it is approximately 6 pm closer to the hydrogens).

Modeling Liquid Water Dynamics: Molecular Dynamics

Molecular dynamics (MD) is the term given to the process of simulating the classical motion of atoms. “Simulation” in this case refers to using a computer program to perform the calculations, “classical” refers to systems obeying Newton's laws (rather than quantum mechanics), and “motion” refers to how the position of atoms changes over time.

The primary characteristic of a molecular dynamics simulation is the “force field,” a collection of potential energy terms that dictate how atoms behave over time. The connection of the force field to one-dimensional classical mechanics is illustrated as follows:

Newton’s Second Law:

$$\vec{F} = m\vec{a} \tag{2}$$

Acceleration is the second time derivative of position:

$$\vec{a} = \frac{d^2\vec{x}}{dt^2} \tag{3}$$

Conservative forces (a class of forces that includes charge interactions) are the negative derivative of the potential energy function:

$$\vec{F} = -\frac{dV}{d\vec{x}} \tag{4}$$

Substituting Eq. 3 and 4 into Eq. 2 gives:

$$-\frac{dV}{d\vec{x}} = m\frac{d^2\vec{x}}{dt^2} \tag{5}$$

Equation 5 shows that the derivative of potential energy (the force field) is related to the second derivative of the position (the motion). The basic form of the force field typically contains terms for bonds and angles (modeled as harmonic oscillators), torsions (modeled with a $\cos\theta$ function), and, importantly for this lab, terms for the electrostatic and van der Waals interactions. The electrostatic interaction between a pair of atoms is described by Coulomb's law:

$$V = \sum_{i=1}^{N}\sum_{j=i+1}^{N}\frac{q_i q_j}{4\pi\epsilon_0 r_{ij}} \tag{6}$$

where *i* and *j* are the atom pair, r_{ij} is the distance between the two atoms, q_i is the charge on atom *i*, q_j is the charge on atom *j* and ε_0 is the permittivity of free space. The Coulombic attraction between the oxygen and one hydrogen atom in water is shown in

Figure 3.

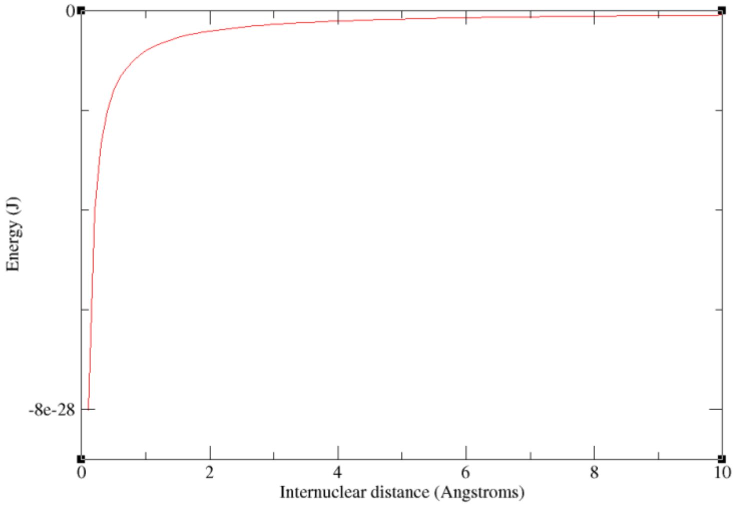


Figure 3. Coulombic interaction between O (*q* = -0.8434e) and H (*q* = 0.417e) in TIP3P water

The van der Waals interactions are non-bonded, non-Coulombic interactions that model dispersion forces between atoms. A common functional form for these interactions is the Lennard-Jones 12-6 potential, which has the form:

$$V = \sum_{i=1}^{N} \sum_{j=i+1}^{N} \left(4\,\epsilon_{ij} \left[\left(\frac{\sigma_{ij}}{r_{ij}} \right)^{12} - \left(\frac{\sigma_{ij}}{r_{ij}} \right)^{6} \right] \right) \tag{7}$$

where ε_{ij} is the depth of the well and σ_{ij} is the collision diameter, as shown in Figure 4. Physically, the large positive (repulsive) energy at small internuclear distances reflects that the atoms have volume and cannot overlap each other. The negative (attractive) energy at longer distances describes the induced dipole-dipole interaction (*i.e.* the dispersion force).

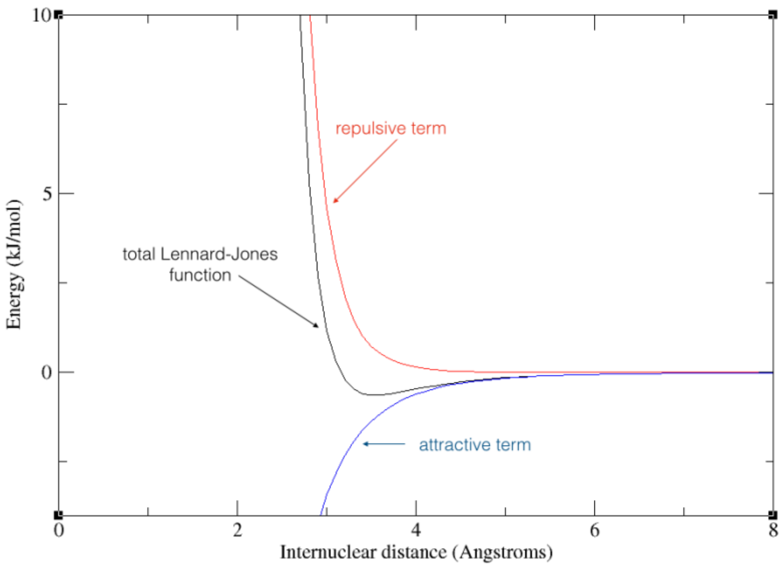


Figure 4. Lennard-Jones function of TIP3P water (σ = 3.15061Å, ε = 0.6364 kJ/mol)

As seen in the electrostatic and van der Waals terms, the force field uses parameters for each atom type in the system. In larger molecules, there are multiple types of each atom. For example, an alpha-carbon is treated as a distinct atom from a beta-carbon in the side chain of an amino acid. Thus, in order to run an MD simulation, two sets of information are necessary: first, the parameter (all of the σ, ε, q, etc values) and topology (which atoms are connected to which) file, and secondly, the coordinate file (containing the initial position of each atom). The "Step 0: Preparing the system" section describes how to create these files in the MD program Amber.[7]

Modeling the Water Molecule: TIP3P

A perfect model of bulk liquid water would use quantum mechanics to exactly solve the system's Schrödinger equation, an as of yet insurmountable task. Instead, assumptions with varying degrees of severity are used to create water models designed for different purposes. One general type of water model, using classical mechanics rather than quantum, describes water molecules with point charges (for the Coulombic interactions) and van der Waal parameters (for the dispersion interactions) and is sufficient for applications which do not require the dissociation of the molecule or creation of bonds. This kind of classical model is also significantly faster computationally than solving the Schrödinger equation.

An ubiquitous classical water model is TIP3P,[8] a "3-point" water model that has a point charge on each atom (hence, three points). Only the oxygen atom has parameters for the Lennard-Jones function; van der Waals interactions for the hydrogen atoms are ignored. The atomic charges and Lennard-Jones parameters are given in Figures 3 and 4 respectively. TIP3P water has an O-H bond length of 0.9572 Å, and a H-O-H angle of 104.52 degrees. The bond length and angle are not allowed to change during a simulation: the molecule is kept rigid to make calculations faster in an MD simulation.

Predictions

To recap, the three research questions are:

1. What is the structure of liquid water?
2. How do the dynamics of liquid water change with temperature?
3. How well does your model of liquid water and set of assumptions match experimental results for dynamics and structure?

Based on the background information in the previous section, what information do you need to answer each question? **What is your hypothesis for each? Predict trends for the first two questions as temperature increases, and how well you think TIP3P will recreate experimental values for the third question.**

Procedure

Before beginning the lab, each group should create a folder on the computer desktop for their work (do not include spaces in the folder name). Open the terminal application and navigate to your folder:

```
cd Desktop/YourFolderName
```

This folder will contain all of the files you use during this lab, and serves as a digital record of your work (*i.e.* instead of a paper notebook). Use a text file saved to that folder to record your predictions and contain any additional notes. Copy the contents of the `MD_files` folder into your folder. An annotated list of the files is given in the appendix.

You'll be starting from an ice-h crystal structure. Normally MD simulations begin with a box of liquid-like water, but starting with ice gives the same results, and lets you see how you can go from a solid structure to the liquid structure and dynamics. The structure is given as a PDB file, which was designed for protein crystal structures (PDB stands for "Protein DataBank"). PDB structures of ice are constructed from experimental x-ray diffraction data.[9]

Double-clicking the `ice.pdb` file will open it in the program Chimera,[10] which allows you to see the structure (you can zoom in/out and rotate to see how the polymorph looks from other angles).

An overview of the following steps is given in the flow chart in Figure 5.

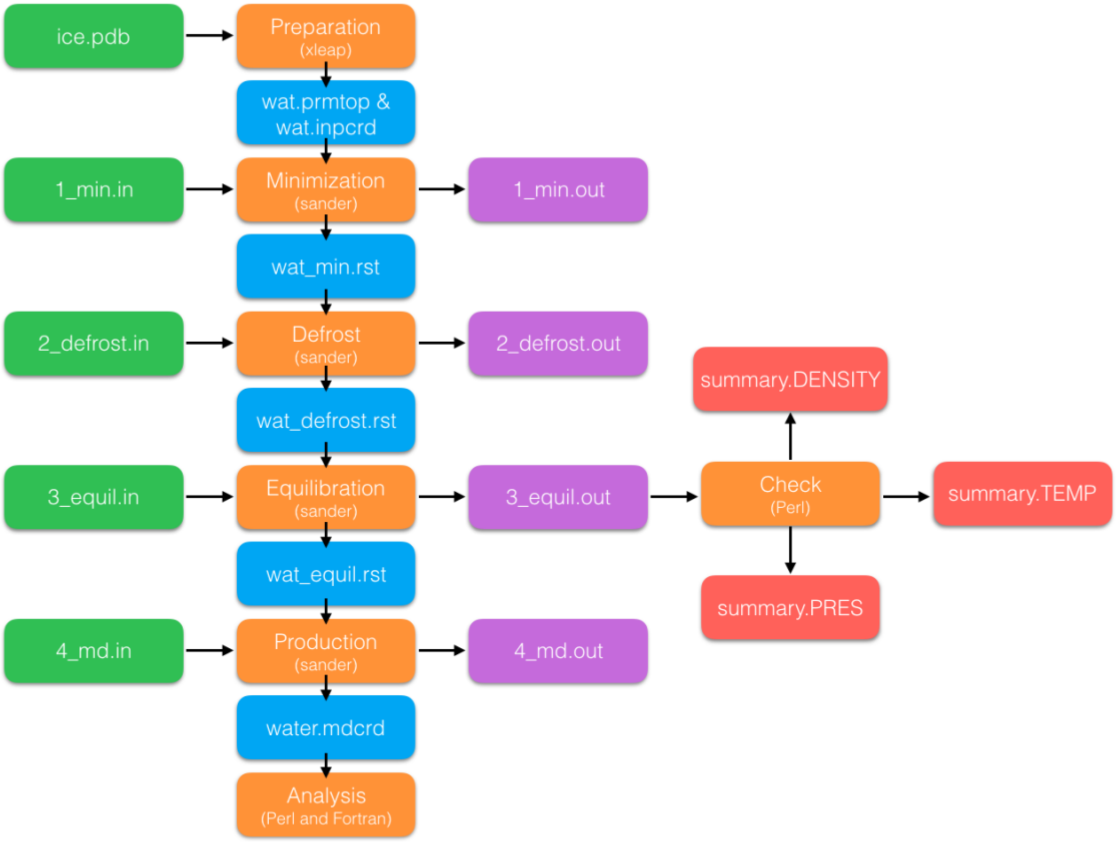


Figure 5. Flow chart of the job input files (green), structure files (blue), programs (orange), output files (purple) and files used to check equilibration (red) used in the procedure.

Step 0: Preparing the system

In order to use the molecular dynamics program Amber, you must use the following command in your terminal window (this is done any time you open a new terminal window):

```
export AMBERHOME=/CLI/amber12
```

***NOTE**: All of the commands to run Amber are given in the `input_commands` file. Copy and paste the commands into the terminal window to avoid typos.*

Amber is actually a collection of subprograms. One subprogram, xleap, is used to prepare systems. To access xleap, enter the following command in your terminal window:

```
$AMBERHOME/bin/xleap -s -f $AMBERHOME/dat/leap/cmd/leaprc.ff99SB
```

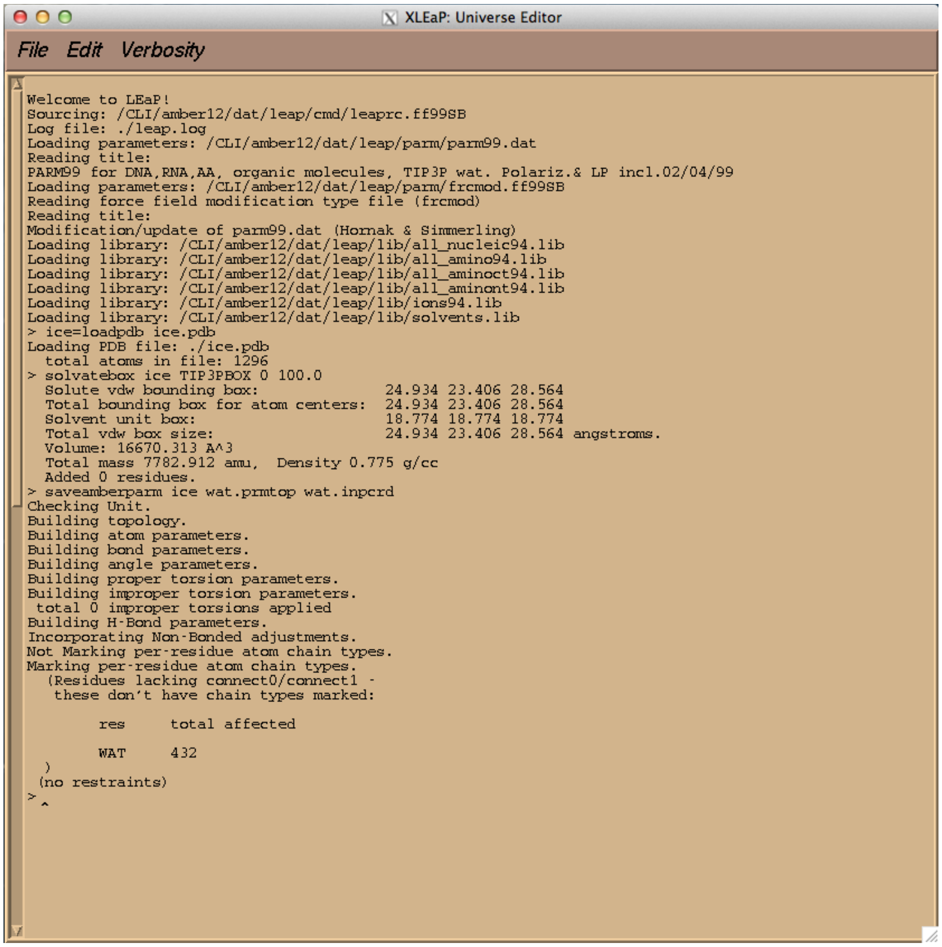
Once xleap opens, you'll import a box of water. Xleap does not allow text to be pasted into its command line, so these must be typed by hand. Importing the box of water is a two-step process:

```
ice=loadpdb ice.pdb
```

followed by

```
solvatebox ice TIP3PBOX 0 100.0
```

where the commands are case-sensitive. Your screen should look like Figure 6.



```
Welcome to LEaP!
Sourcing: /CLI/amber12/dat/leap/cmd/leaprc.ff99SB
Log file: ./leap.log
Loading parameters: /CLI/amber12/dat/leap/parm/parm99.dat
Reading title:
PARM99 for DNA,RNA,AA, organic molecules, TIP3P wat. Polariz.& LP incl.02/04/99
Loading parameters: /CLI/amber12/dat/leap/parm/frcmod.ff99SB
Reading force field modification type file (frcmod)
Reading title:
Modification/update of parm99.dat (Hornak & Simmerling)
Loading library: /CLI/amber12/dat/leap/lib/all_nucleic94.lib
Loading library: /CLI/amber12/dat/leap/lib/all_amino94.lib
Loading library: /CLI/amber12/dat/leap/lib/all_aminoct94.lib
Loading library: /CLI/amber12/dat/leap/lib/all_aminont94.lib
Loading library: /CLI/amber12/dat/leap/lib/ions94.lib
Loading library: /CLI/amber12/dat/leap/lib/solvents.lib
> ice=loadpdb ice.pdb
Loading PDB file: ./ice.pdb
  total atoms in file: 1296
> solvatebox ice TIP3PBOX 0 100.0
  Solute vdw bounding box:              24.934 23.406 28.564
  Total bounding box for atom centers:  24.934 23.406 28.564
  Solvent unit box:                     18.774 18.774 18.774
  Total vdw box size:                   24.934 23.406 28.564 angstroms.
  Volume: 16670.313 A^3
  Total mass 7782.912 amu,  Density 0.775 g/cc
  Added 0 residues.
> saveamberparm ice wat.prmtop wat.inpcrd
Checking Unit.
Building topology.
Building atom parameters.
Building bond parameters.
Building angle parameters.
Building proper torsion parameters.
Building improper torsion parameters.
 total 0 improper torsions applied
Building H-Bond parameters.
Incorporating Non-Bonded adjustments.
Not Marking per-residue atom chain types.
Marking per-residue atom chain types.
  (Residues lacking connect0/connect1 -
   these don't have chain types marked:

        res     total affected

        WAT     432
  )
 (no restraints)
> ^
```

Figure 6. Demonstration of the correct xleap commands and output.

Once you've created your box of waters (congratulations!), you create the parameter and topology file (.prmtop) and initial coordinate file (.inpcrd) by typing into xleap the command:

```
saveamberparm ice wat.prmtop wat.inpcrd
```

Once the files have been saved, you can type the command `quit` to exit xleap.

Step 1: Energy minimization

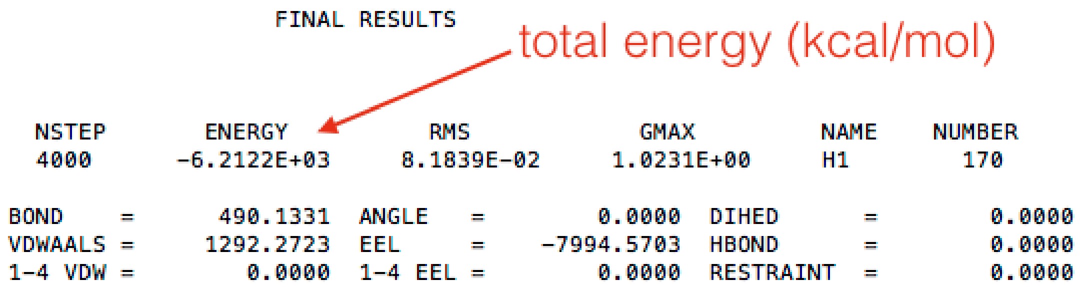
The next four steps will use the molecular dynamics subprogram of Amber: sander. First, we'll need to minimize the energy of the system. This step is done to remove any bad contacts (overlapping atoms). The command to run this step is:

```
$AMBERHOME/bin/sander -O -i 1_min.in -o wat_min.out -p wat.prmtop
                 -c wat.inpcrd -r wat_min.rst
```

The format of the command is:

- -O allows the program to overwrite any existing files
- -i provides the input file, which controls the temperature, pressure, how long the job runs for, etc
- -o provides the output file, which contains energy, temperature, pressure, volume, density, etc information for each time step (printed as often as determined by the .in file)
- -p provides the parameter and topology file
- -c provides the input coordinate file
- -r provides the restart file, which will contain the final coordinates of the minimization (and will be the initial coordinates for step 2).

This step should only take about a minute to run. Once it's done, open the `wat_min.out` file in a text editor and compare the total energy of the first timestep to that of the final timestep and make sure the energy has gone down. Figure 7 demonstrates where to find the energy value in these files.



```
                              FINAL RESULTS



   NSTEP       ENERGY          RMS            GMAX         NAME    NUMBER
   4000     -6.2122E+03     8.1839E-02     1.0231E+00     H1        170

 BOND    =      490.1331  ANGLE   =        0.0000  DIHED      =        0.0000
 VDWAALS =     1292.2723  EEL     =    -7994.5703  HBOND      =        0.0000
 1-4 VDW =        0.0000  1-4 EEL =        0.0000  RESTRAINT  =        0.0000
```

Figure 7. Example of readout from minimization output file.

Step 2: Defrost

After the bad contacts are removed, we need to defrost (melt) the system. Because we currently only have positions for the atoms (and no velocities), the system is at absolute zero. Open the `2_defrost.in` file and **change the "`temp0   = 300.0`," line to your assigned temperature (in K)**.

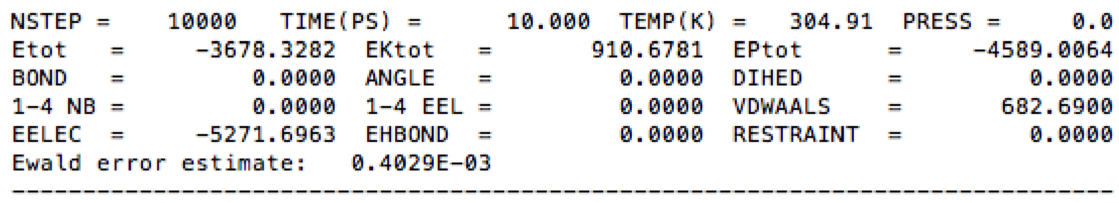
Table 1: Range of temperatures for simulation of liquid water

Simulation Number	Temperature (K)
1	280
2	290
3	300
4	310
5	320
6	330
7	340
8	350
9	360
10	370

Save the file and run the defrost with the command:

```
$AMBERHOME/bin/sander -O -i 2_defrost.in -o wat_defrost.out -p
         wat.prmtop -c wat_min.rst -r wat_defrost.rst
```

This step takes about 2 minutes to run. Note that the -c flag now references the .rst file from the first step. Open the `wat_defrost.out` file and check the temperature at the final timestep; it should be close to your target temperature (it probably won't be exact, as the temperature fluctuates around the target value). An example of the defrost output file is given in Figure 8.



```
NSTEP =    10000   TIME(PS) =      10.000  TEMP(K) =   304.91  PRESS =     0.0
Etot   =     -3678.3282  EKtot   =       910.6781  EPtot      =     -4589.0064
BOND   =         0.0000  ANGLE   =         0.0000  DIHED      =         0.0000
1-4 NB =         0.0000  1-4 EEL =         0.0000  VDWAALS    =       682.6900
EELEC  =     -5271.6963  EHBOND  =         0.0000  RESTRAINT  =         0.0000
Ewald error estimate:   0.4029E-03
------------------------------------------------------------------------------
```

Figure 8. Example of readout from defrost output file.

Step 3: Equilibration

Once the system is at the correct temperature, it must be brought to equilibrium. Practically, it means that the density must be allowed to reach a steady value (ideally around 1 g/cm^3) for your temperature and a pressure of 1 atm. Open the `3_equil.in` file and change the "`tempi = 300.0, temp0 = 300.0`," line with your target temperature value. Run the equilibration step with the following command:

```
    $AMBERHOME/bin/sander -O -i 3_equil.in -o wat_equil.out -p
          wat.prmtop -c wat_defrost.rst -r wat_equil.rst
```

This step will take about 10 minutes. Once it is done, you'll process its output to ensure you've reached equilibrium. You'll need to create a folder to hold files that will be created by running the `process_mdout.perl` script. Do so using these commands:

- `mkdir EQUIL`
  - this command creates the EQUIL folder
- `cd EQUIL`
  - this command moves you into the EQUIL folder
- `cp ../process_mdout.Perl .`
  - this command copies the Perl script from the folder above the EQUIL folder
- `Perl process_mdout.Perl ../wat_equil.out`
  - this command runs the Perl script inside the EQUIL folder

Use xmgrace to open the `summary.PRESS, summary.TEMP`, and `summary.DENSITY` files to check that the pressure is fluctuating around 1 atm (it will have very large fluctuations), that the temperature is fluctuating around your target value, and that the density has plateaued at a value near 1 g/cm^3. Figure 9 shows an example of the `summary.DENSITY` file; note that the density rose and then leveled off over the course of the simulation. After you are done checking the equilibrium, use the following command to go back to the folder where you are running Amber:

```
cd ..
```

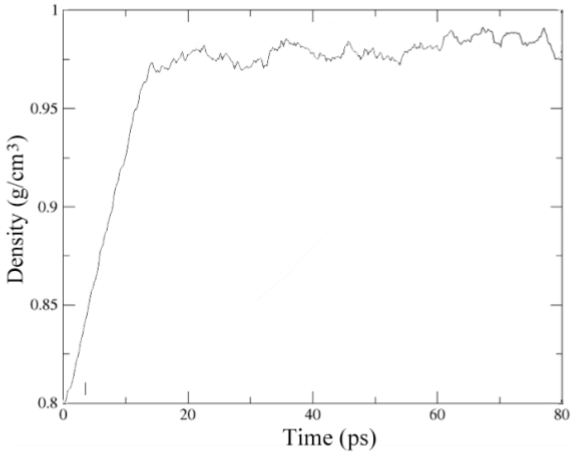


Figure 9. Density (g/cm^3) over the course of the 80 ps equilibration.

Step 4: Production

Once the system is properly equilibrated, we can begin collecting data. Open the

`4_md.in` file and change the temperature to your target value. While the file is open, take a look at a few important settings:

- `pres0 = 1.0`
  - this flag sets the pressure to 1 atm.
- `tempi = 300.0, temp0 = 300.0,`
  - as you've seen, these flags set the initial and final temperatures (and are the same as each other unless you're defrosting). Make sure these temperatures are changed to your assigned temperature value.
- `nstlim = 20000`
  - the total number of time steps the simulation will use, and combined with:
- `dt = 0.001`
  - the difference in time for each timestep (time is measured in picoseconds, but the timestep is one femtosecond, which is one thousandth of a picosecond); the total simulation time is 20 ps.
- `ntwx = 1`
  - this flag tells Amber to write out the coordinates to a trajectory file every timestep. Having it set to 1 means we'll end up with 20,000 steps worth of coordinates in the trajectory for analysis (which, given that you should have 432 waters total, and each water has 3 atoms, and each atom has 3 coordinates, gives a grand total of 77,760,000 coordinates in the trajectory file!)

The production simulation is run using the command:

```
$AMBERHOME/bin/sander -O -i 4_md.in -o wat_md.out -p wat.prmtop -
          c wat_equil.rst -r wat_md.rst -x water.mdcrd
```

and takes about 5 minutes to run. This is faster than the equilibration because that job ran 40,000 steps with a *dt* of 2 fs, for a total of 80 ps of data. Even if you're not simulating a long production, you always have to make sure the system is completely equilibrated first. The `-x` flag in the commend provides the name of the trajectory file, in this case `water.mdcrd`. It's this file that will be analyzed in the post-processing part of the lab.

Analysis

Viewing the equilibrated system

Create a PDB file of a snapshot (single moment in time) of the equilibrated system, using the command

```
$AMBERHOME/bin/ambpdb -p wat.prmtop < wat_md.rst > wat_equil.pdb
```

Open the file `wat_equil.pdb` in Chimera to view the water molecules. **How is the system different than the original `ice.pdb`?**

Compiling analysis codes

NOTE: If they have not yet been compiled, you will need to run a separate command (taken from input_commands.txt) to compile each of the following codes:

- parse_mdxyz.f90
- diffusion.f90
- grOO.f90

Preparing the coordinate file

From the molecular dynamics simulation, you have the file `water.mdcrd` that contains the *x*- , *y*- , and *z*-coordinates of every atom in 432 water molecules for 20 ps of simulated motion. In order to learn anything meaningful from this dynamics data, we'll use other code to extract what we need from that file and perform calculations on the data. The first step is to rewrite the Amber trajectory into a different format. If you open the `water.mdcrd` file in a text editor, you can see that there are 10 columns of *x*- , *y*- , and *z*-coordinates. You'll use a Perl script to create a file (`water.mdxyz`) that has each coordinate on its own line:

```
perl parse_mdcrd.pl
```

Figure 10 shows a flow chart of the process used for analyzing your MD data.

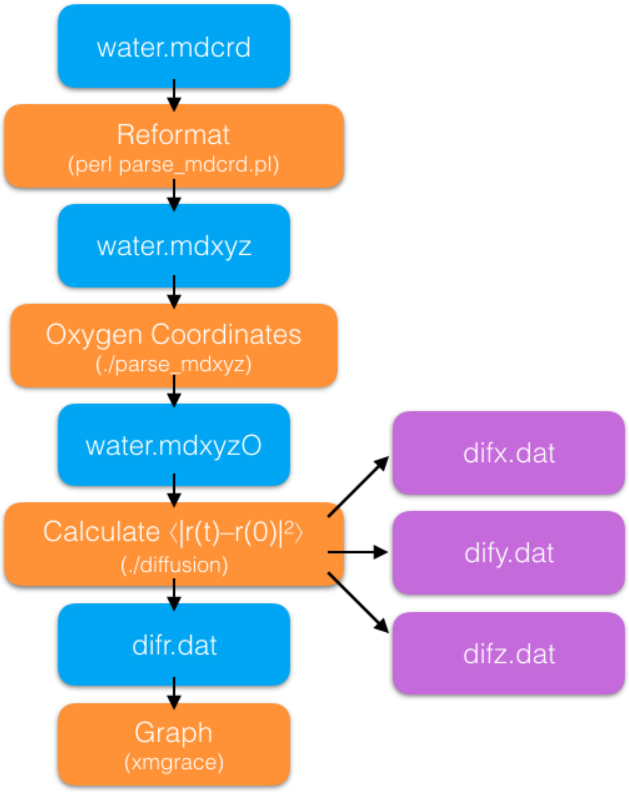


Figure 10. Flow chart of the files and code used in the analysis.

Radial distribution function

Although it is possible to calculate all three RDFs (OO, OH, HH) from the MD data, analyzing the just the $g(r)_{OO}$ is sufficient to answer the research questions. In order to generate the OO RDF, you'll use the `grOO` Fortran code. It requires the `water_box.mdxyz` file that is created by the `parse_mdxyz` code. The `grOO` code calculates how many oxygen atoms are in concentric spherical volumes surrounding a chosen oxygen, compared to the average density of the water. Use the command

```
./grOO
```

to run the code and use

```
xmgrace grOO.dat &
```

to plot the results.

You should determine the position and intensity of the first maximum and the position of the first minimum for the $g(r)_{OO}$ at each temperature. You can find these values by double clicking the screen to open the `Set Appearance` window, then right-clicking the data set and choosing to `Edit in Spreadsheet`.

Diffusion coefficient

You'll need to use two Fortran codes for the calculation of the diffusion coefficient. The first, `parse_mdxyz`, takes the `water.mdxyz` file and pulls out just the coordinates of the oxygen atoms, because we're tracking the movement of the water molecule as a whole by the movement of the oxygen atom. The code is run using the command

```
./parse_mdxyz
```

(both Fortran codes assume that you have 432 waters, that the trajectory file contains 20,000 steps and that you've already run the `parse_mdcrd.pl` file to create `water.mdxyz`).

Now you can calculate the quantity $\langle|\vec{r}_i(t) - \vec{r}_i(0)|^2\rangle$ using the command

```
./diffusion
```

to run the second Fortran code. It will create four new files in your folder: difr.dat, difx.dat, dify.dat, and difz.dat. The latter three files contain the *x*-, *y*-, and *z*-component data while the difr.dat file contains the magnitude of the vector data (and is thus what we want to graph).

Use `xmgrace` to plot difr.dat (plotting $\langle|\vec{r}_i(t) - \vec{r}_i(0)|^2\rangle$ vs timesteps). **Use the plot to determine the diffusion coefficient *D* for your target temperature.** Note that the slope will be in terms of timesteps, so you'll have to do a conversion to get your *D* value in the proper units of Å^2/ps.

Conclusions

In order to draw conclusions from these results, you need to answer each research question. Additional questions and resources to consider for each have been provided. Answer these questions in the same text file as your predictions.

1. What is the structure of liquid water?
    a. Consider the trend, if any, for the position and intensity of the first peak and the position of the first minimum as water temperature increases. Does this trend match your prediction? What is the physical significance of this trend?
2. How do the dynamics of liquid water change with temperature?
    a. Consider the trend, if any, in diffusion coefficients as water temperature increases. Does this trend match your prediction? What is the physical significance of this trend?
3. How well does your model of liquid water and set of assumptions match experimental results for dynamics and structure?
    a. Look at the results for 300 K, the most commonly reported RDFs and *D*. Compare these results to those in the literature for (real) water: use reference 2 for RDF comparisons and reference 3 for *D* comparisons. For the RDF, you may need to compare the position and height of the first peak. Does TIP3P match either of these values (structure or dynamic) for real water, or was one more clearly prioritized by the modelers who created TIP3P?
    b. Using the literature (Mark, P.; Nilsson, L. Structure and Dynamics of the TIP3P, SPC, and SPC/E Water Models at 298 K. *J. Phys. Chem. A.* **2001**, *105*, 9954-9960), compare TIP3P to another water model, SPC/E. How do the priorities of these water models differ?
    c. Is there an application where it makes more sense to use TIP3P than SPC/E?

Appendix

Annotated List of Files

- `1_min.in` -- input file for the minimization step of the molecular dynamics run
- `2_defrost.in` -- input file for the defrost (heating) step of the molecular

dynamics run

- `3_equil.in` -- input file for the equilibrium step of the molecular dynamics run
- `4_md.in` -- input file for the production step of the molecular dynamics run
- `diffusion.f90` -- the Fortran code for the diffusion calculation (must be compiled)
- `grOO.f90` -- the Fortran code for the radial distribution function calculation (must be compiled)
- `ice.pdb` -- the file containing atom types and coordinates for the initial ice structure. Contains 432 water molecules
- `input_commands.txt` -- a list of commands used to run the molecular dynamics jobs, compile the Fortran code, and run the executables created by the compiling step
- `parse_mdcrd.pl` -- Perl script that reformats the trajectory file from the molecular dynamics run
- `parse_mdxyz.f90` -- Fortran code that extracts the oxygen coordinates from the reformated trajectory (must be compiled)

Bibliography

1. D. Eisenberg and W. Kauzmann, *The Structure and Properties of Water.* Oxford University Press (1969).
2. A. Soper. Journal of Chemical Physics, 258: 121 (2000).
3. R. Mills. *Journal of Physical Chemistry*, 77: 685 (1973).
4. K. R. Harris, B. Gambold, and W. S. Price. *Journal of Chemical and Engineering Data*, 60: 3506 (2015).
5. F. Franks, editor, *Water: A Comprehensive Treatise.* Plenum Press (1972).
6. M. Allen and D. Tildesley, *Computer Simulations of Liquids*. Oxford University Press, Inc. (1987)
7. D. A. Case, T. A. Darden, T. E. Cheatham, C. L. Simmerling, J. Wang, R. E. Duke, R. Luo, R. C. Walker, W. Zhang, K. M. Merz, B. Roberts, S. Hayik, A. Roitberg, G. Seabra, J. Swails, A. W. Goetz, I. Kolossváry, K. F. Wong, F. Paesani, J. Vanicek, R. M. Wolf, J. Liu, X. Wu, S. R. Brozell, T. Steinbrecher, H. Gohlke, Q. Cai, X. Ye, J. Wang, M. J. Hsieh, G. Cui, D. R. Roe, D. H. Mathews, M. G. Seetin, R. Salomon-Ferrer, C. Sagui, V. Babin, T. Luchko, S. Gusarov, A. Kovalenko, P. A. Kollman. Amber12 (2012)
8. W. Jorgensen, J. Chandrasekhar, J. Madura, R. Impey and M. Klein. *Journal of Chemical Physics*, 79: 926 (1983).
9. P. Owston. *Acta Crystallographica*, 2: 222 (1949)
10. E.F. Pettersen, T. D. Goddard, C. C.Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, T. E. Ferrin. *Journal of Computational Chemistry.* 13: 1605 (2004)