Lab2:

- Introduction – brief summary and reason for project
- Procedure – what steps did you actually perform and in what order
- Methods – kinds of calculations, programs used, level of theory, any assumptions or approximation such as quantities varied, restraints and constraints, special program options
- Results – data obtained, usually in tables and plots
- Analysis – explain your result

Chemistry 430 — Simulation in Chemistry & Biochemistry Laboratory #2 — Properties of Pure Liquids via Molecular Dynamics Simulation In this lab you will prepare a cubic periodic box containing a pure organic liquid, and then generate a molecular dynamics trajectory for the liquid. Post-processing analysis of the trajectory will then give estimates of the density, heat of vaporization, radial distribution, diffusion constant, etc. for your chosen liquid.

Protocol

(1) Below is a table containing the molecular weight, density, melting point and boiling point for several simple liquids. Your goal in this lab is to pick one of these molecules, perform a molecular dynamics simulation on a liquid sample of your chosen substance, and then compute values for the density, heat of vaporization, diffusion constant and radial distribution functions. For the molecules that exist in multiple conformations, you can also monitor the approach to conformational equilibrium.

(2) First, obtain a TINKER .xyz file with a single molecule of your chosen substance. Files for isolated molecules, set up to use the OPLS-AA force field, are provided on the course website for this lab. You will need to use the oplsaa.prm file also provided on the lab site. This version of the parameter file has extra parameters for some molecules in the list. In addition, copy the tinker.key file from the course site, which initially contains just a single line pointing to the OPLS-AA parameters. Place all of these files into a new directory that you create and where you will perform this lab.

glycol.xyz

Molecule	MW	Density	MP(wrt 273 K)	BP (wrt 273 K)
Ethylene Glycol	62.07	1.113	-13	197

(3) Examine the .xyz file and compare the atom types in the column just to the right of the x,y,z-coordinates against the OPLS-AA parameter file obtained in the previous step to verify that the correct atom types are used for your molecule.

```
analyze glycol.xyz E
```

Total Potential Energy : 44.5524 Kcal/mole

(4) Minimize your molecule using the Tinker minimize program to clean up any distortions in the original geometry. Copy your molecule to a second file with some different name, edit this second file to translate the molecule to some new set of coordinates using any text editor, and then use the Tinker xyzedit program to merge your two molecule files into a single file containing a dimer. Run the minimize program to find the optimal structure and energy for the dimer. What is the energy of interaction of the two molecules in the dimer, i.e., the energy of the dimer minus twice the energy of the monomer? Are you sure that you have found the best dimer structure?

```
minimize glycol.xyz .01
analyze glycol.minimized.xyz E
```
Total Potential Energy : 2.3237 Kcal/mole

(12) Translate All Atoms by an X,Y,Z-Vector
```
printf "12\n6 6 6\n" | xyzedit glycol.minimized.xyz && mv
glycol.minimized.xyz_2 glycol.minimized_translated.xyz
```

(22) Append a Second XYZ File to Current One
```
printf "22\nglycol.minimized_translated.xyz" | xyzedit glycol.minimized.xyz
&& mv glycol.minimized.xyz_2 glycol.minimized_2_molecules.xyz
```

```
analyze glycol.minimized_2_molecules.xyz E
```
Total Potential Energy : 4.5959 Kcal/mole

```
minimize glycol.minimized_2_molecules.xyz .01 && mv
glycol.minimized_2_molecules.xyz_2 glycol.minimized_2_molecules_minimized.xyz
analyze glycol.minimized_2_molecules_minimized.xyz E
```
Total Potential Energy : -3.4043 Kcal/mole

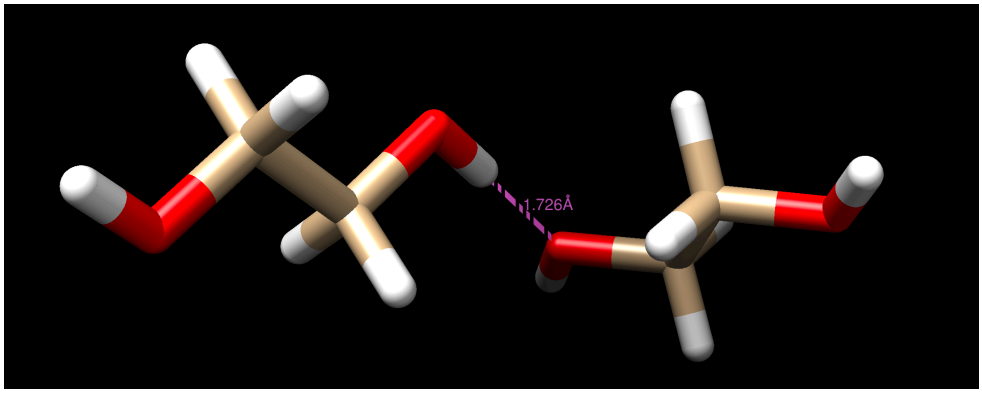
-3.4043 -2*(2.3237) = -8.0517Kcal/mole

Form H-bond at 1.7 angstroms. Probable not best dimer as there should be able to create 2 hydrogen bonds on each hydroxyl group. Pretty sure there is no way a minimized structure would occur with all 4 hydrogen bonds (2 on each -OH) group as not sure if hydrogen bonds have any energy constraints on crossing planes (sounds like it's not even a thing) plus there would be clashes on the other atoms.

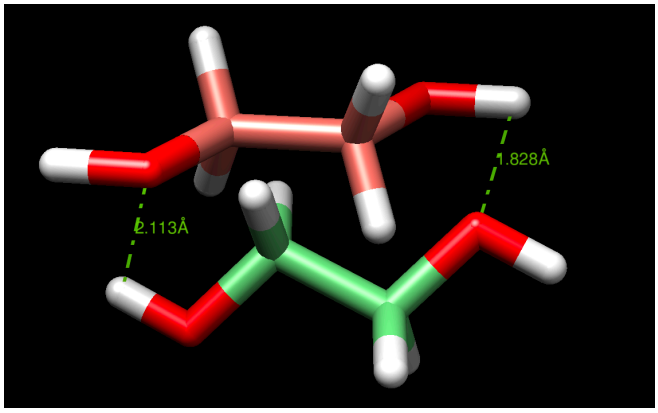
Function to convert Tinker xyz file to .sdf file for Chimera visualization (won't work for arc files and for every case, requires obabel)

```
function TSDF {
```

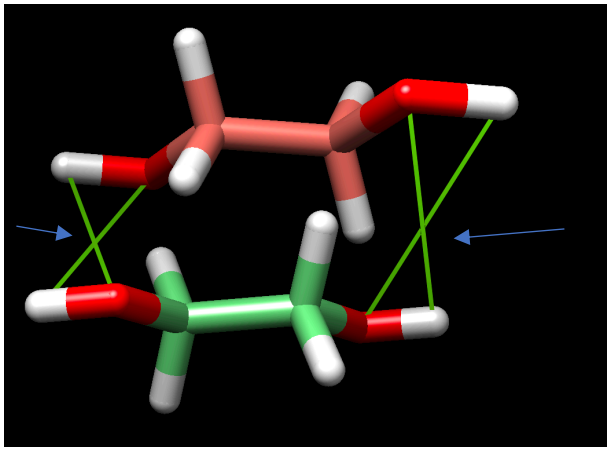
```
    input=$1
    # get line number of first atom
    first_mol_line=$(grep -n "1 " $input | head -n1 | cut -d: -f1)
    (head -n $((first_mol_line-1)) $input;
    awk -v line=$first_mol_line 'FNR >= line && $2=substr($2,1,1) { print }'
OFS='\t' $input) > test.xyz
    obabel -itxyz -osdf test.xyz -O $(dirname $input)/$(basename $input
.xyz).sdf
    rm test.xyz
}

TSDF glycol.minimized_2_molecules_minimized.xyz
```



??



Is it possible for H-bonds to cross?



(5) Run the Tinker xyzedit program on the minimized monomer (not the dimer from step 4!). Use the xyzedit option to create a cubic periodic box filled with multiple copies of your molecule. You should construct a box of about 25-30 Angstroms on a side, and you will need to calculate the number of molecules to place into the box based on the molecular weight and density of your chosen liquid. Note that 1 Angstrom is equal to 10^-10 meters or 0.1 nanometers, and Avogadro's number is 6.02214076 x 10^23.

6.02214076e23 molecules/mole / 62.07 g /mole * 1.11 g/cm^3 * 30 angtroms^3
30 angstroms = 30e-8 cm
30 angtroms^3 = (30e-8 cm)^3
6.02214076e23 molecules/~~mole~~ / 62.07 ~~g~~ /~~mole~~ * 1.11 ~~g~~/~~cm^3~~ * (30e-8^3 ~~cm)^3~~

6.02214076e23 molecules / 62.07 * 1.11 * (30e-8)^3 ~ 291 molecules

```
printf "23\n291\n30,30,30\nY" | xyzedit glycol.minimized.xyz && mv
glycol.minimized.xyz_2 glycol.minimized.box.xyz
```

(6) Make sure the box size in Angstroms is set via the appropriate keyword (for example, X-AXIS 25.0, or whatever box size you chose in step 5) in your tinker.key keyfile. In that same keyfile, you should also set the cutoff distance for the van der Waals interactions. This value must be **somewhat less than half** the box dimension – a value of 9 Angstroms is typical for OPLS simulations. For example, you could add the keyword VDW-CUTOFF 9.0. We will use Ewald summation for electrostatics by adding the **EWALD** keyword to the keyfile. And set the real-space Ewald cutoff to 7 Angstroms via **EWALD-CUTOFF** 7.0 keyword. Then enable use of pair neighbor lists for both VDW and electrostatic interactions by adding the **NEIGHBOR-LIST** keyword. Finally, add the RATTLE keyword to fix all bonds to hydrogen atoms at their ideal lengths.

- NEIGHBOR-LIST Turns on pairwise neighbor lists for partial charge electrostatics, atomic multipole electrostatics, induced dipole polarization and any of the van der Waals potentials. This method will yield identical energetic results to the standard double loop method.

- EWALD Turns on the use of smooth particle mesh Ewald (PME) summation during computation of partial charge, atomic multipole and polarization interactions in periodic systems. By default, in the absence of the EWALD keyword, distance-based cutoffs are used for electrostatic interactions.

- **D**EWALD Turns on the use of smooth particle mesh Ewald (PME) summation during computation of **d**ispersion interactions in periodic systems. By default, in the absence of the DEWALD keyword, distance-based cutoffs are used for dispersion interactions.
    - D for London dispersion forces

- RATTLE [BONDS / ANGLES / DIATOMIC / TRIATOMIC / WATER] Invokes the rattle algorithm, a velocity version of shake
    - HANS C. ANDERSEN
        - On computers of fixed precision, it is of higher precision than SHAKE. Since it deals directly with the velocities, it is easier to modify RATTLE for use with the recently (recently ...funny) developed constant temperature and constant pressure molecular dynamics methods and with the nonequilibrium molecular dynamics methods that make use of rescaling of the atomic velocities.

glycol.minimized.box.xyz

(7) Next you will minimize the energy of the solvent box to remove any bad or high energy contacts. To complete this step, run the Tinker minimize program to perform the minimization of your liquid box. You should use a moderate convergence criterion for the **minimization of about 1.0 kcal/mol/Ang instead of the tighter default of 0.01**. If your initial attempt at minimization fails, you may need to first perform the minimization with the CHARGETERM NONE keyword line added to the tinker.key file, then rerun the minimization after removing this keyword.

1 angstrom = 10 nm
1.0 kcal/mol/Ang * 10nm/Ang = 10 kcal/mol/nm = 41.84 Kj/mol/nm

work (joules) = force (newtons) x distance (meters),
work (joules) / distance (meters) = force (newtons)

Energy pre-minimization
```
analyze glycol.minimized.box.xyz $(echo E)
```
123.3423 Kcal/mole

```
minimize glycol.minimized.box.xyz 1.0 && mv glycol.minimized.box.xyz
glycol.minimized.box_minimized.xyz
```

Energy post-minimization

```
analyze glycol.minimized.box_minimized.xyz $(echo E)
```

-3865.5692 Kcal/mole !!

(8) Start an MD simulation using the Tinker dynamic program and performed in the isothermal-isobaric ensemble (constant temperature and pressure, also referred to as NPT). We will use a standard integration method due to Beeman, and add a thermostat and barostat to maintain temperature and pressure, respectively. First, put the keyword INTEGRATOR BEEMAN into the tinker.key file. Then, when running the calculation, use room temperature (298K = 25°C) for the target temperature, as long as your substance is a liquid at that temperature. Otherwise use a target temperature midway between the melting and boiling points. The target pressure should be 1 Atm.

(9) As input to the dynamic, you will be asked for values related to collection of your MD trajectory. Use a 2 fs (fs = femtosecond = 10^-15 seconds) time step for integration, and save coordinate snapshots every 1.0 ps (ps = picosecond = 10^-12 seconds). Ideally, your production period should be run for at least a nanosecond (10-9 seconds) or longer. **You will generally need to discard the first portion of the trajectory (perhaps 200 ps)** as an "equilibration period". Following equilibration, you can use the remaining "production period" of the trajectory in your analysis to determine the density, heat of vaporization, and radial distributions for the OPLS-AA model of your liquid. The full trajectory produced by dynamic will be written to a single Tinker archive (.arc) file.

(10) In order to actually run your MD calculation, you should submit the job in the "background" on your computer inside a terminal window. This can be done with a command similar to the one below, which incorporates the suggested options from steps 8 and 9 above: dynamic mol.xyz 500000 2.0 1.0 4 298.0 1.0 >& mol.log & where you should replace "mol" in this command with the file name you have chosen for your system. Make sure you understand all the parts of this command. Issuing such a command will start the MD job running, send any output that would have displayed on the screen to a file named "mol.log", and return a command line prompt in your terminal window. The MD job will run from some time (perhaps an hour), so you will probably want to continue with the rest of the lab at some later time.

dynamic mol.xyz 500000 2.0 1.0 4 298.0 1.0 >& mol.log &

dynamic <filename> <# of steps> <Time Step Length in Femtoseconds> <Time between Saves in Picoseconds> <Statistical Mechanical Ensemble> <Temperature in Degrees K> <pressure in atm>

1 pico second = 1000 femto seconds
500000steps*2 femto sec = 1 million femto seconds or 1000 pico seconds
1000 pico seconds / save every 1 pico second = 1000 systems in trajectory (.arc) file
First 200 pico seconds is the first 1/5 or first 200 systems.
For 291 molecules @ 10 atoms each plus 1 row for head and 1 row for periodic boundary is 2910+2=2912 per system * 200 systems is 582400 to skip

2912000 – 582400 = 2329600 lines

```
tail -n+582401 glycol.minimized.box_minimized.EWALD.arc | wc -l
tail -n+582401 glycol.minimized.box_minimized.EWALD.arc >
glycol.minimized.box_minimized.EWALD.equilabrated.arc
```

Also need to remove 1/5 of the total steps or 1/5*500000 steps = first 100 to 100000 steps to remove from log file the equilibration stage

```
# remove starting at
grep -n " 100 Dynamics Steps" mol.EWALD.log # line 18
# remove ending at line before this one
grep -n " 100100 Dynamics Steps" mol.EWALD.log # 12018 – 1 = 12017
# remove line 18 to 12017
(head -n17 mol.EWALD.log; tail -n+12018 mol.EWALD.log) >
mol.EWALD.equilibrated.log
```

For liquid

```
dynamic glycol.minimized.box_minimized.EWALD.xyz 500000 2.0 1.0 4 298.0 1.0
>& mol.EWALD.log &
```

For gas phase simulation first prepare 1 molecule in box to simulate at BP
BP is 273K + 197K = 470K

```
printf "23\n1\n30,30,30\nY" | xyzedit glycol.minimized.xyz && mv
glycol.minimized.xyz_2 glycol.minimized.box.for_gas.xyz

minimize glycol.minimized.box.for_gas.xyz 1.0 && mv
glycol.minimized.box.for_gas.xyz_2 glycol.minimized.box.for_gas_minimized.xyz

## BP is 197K over 273K or 470K
dynamic glycol.minimized.box.for_gas_minimized.xyz 500000 2.0 1.0 4 470.0 1.0
>& mol.EWALD.gas.log &
```

(11) Average properties from your MD run can be determined by using the Tinker `mdavg` script (in the /bin directory of the Tinker distribution) on the log file from the dynamics calculation. This will provide the **average temperature, pressure, density, potential energy**, etc. The instantaneous values of these same quantities can be found by inspecting and using grep on the log file. In particular, you should compare your computed density with the experimental value. What is the percentage error in the simulated density?
mdavg mol.EWALD.log

```
mdavg mol.EWALD.log
Total MD Blocks               5000 Blocks
 Total Energy             -819.8518 Kcal/mole    (+/-  62.3479)
 Potential Energy        -2885.9371 Kcal/mole    (+/-  49.0968)
```

```
 Kinetic Energy            2066.0854 Kcal/mole    (+/-  24.8484)
 Temperature                  297.86 Kelvin       (+/-     3.58)
 Pressure                       1.82 Atomsphere   (+/-   233.42)
 Density                      1.0912 Grams/cc     (+/-   0.0084)
```

abs(1.0939 - 1.113)/ 1.113*100 = 1.72% error

```
mdavg mol.EWALD.gas.log
Total MD Blocks               5000 Blocks
 Total Energy               19.8569 Kcal/mole    (+/-   3.7407)
 Potential Energy           10.1107 Kcal/mole    (+/-   1.8967)
 Kinetic Energy              9.7462 Kcal/mole    (+/-   2.0383)
 Temperature                467.09 Kelvin        (+/-    97.69)
 Pressure                    -0.26 Atomsphere    (+/-     2.99)
 Density                    0.0039 Grams/cc      (+/-   0.0000)
```

(12) You should figure out how to compute the heat of vaporization from your MD results. See the Leach textbook, or other recommended course books for further information on how to determine the heat of vaporization.

https://sunxiaoquan.wordpress.com/2015/04/11/calculating-heat-of-vaporization-of-solvent-with-md-simulation/

deltaHvap = Epot(gas)/1 - Epot(liq)/(number of molecules) + RT (ePot of gas for 1st molecule to transition from liquid to gas)

Average Potential Energy -2885.9371 Kcal/mole / 291 molecules (+/- 49.0968)
Average Temperature 297.86 Kelvin

10.1107 Kcal/mole / 1 molecule - (-2885.9371 Kcal/mole / 291 molecules) + RT
10.1107/1 Kcal/mole – (-2885.9371 /291 Kcal/mole) + [1.987cal/K*mole * .001Kcal/cal * 298K]
10.1107- (-9.91731) + 0.5921
deltaHvap = 20.62011 Kcal/mole
(10.1107- (-2885.9371/291) + 0.5921)* 4.184 = **86.27454 Kj/mol**

Maybe I need to run mdavg on the equilibrated portion only to get a more accurate Heat of Vaporization?

```
mdavg mol.EWALD.equilibrated.log
Total MD Blocks               4000 Blocks
 Total Energy              -830.0383 Kcal/mole    (+/-  51.9030)
 Potential Energy         -2896.2594 Kcal/mole    (+/-  34.8049)
 Kinetic Energy            2066.2212 Kcal/mole    (+/-  22.5542)
 Temperature                  297.88 Kelvin       (+/-     3.25)
 Pressure                       0.95 Atomsphere   (+/-   222.09)
 Density                      1.0939 Grams/cc     (+/-   0.0048)
```

gas phase

```
# remove starting at
grep -n " 100 Dynamics Steps" mol.EWALD.gas.log # line 18
# remove ending at line before this one
```

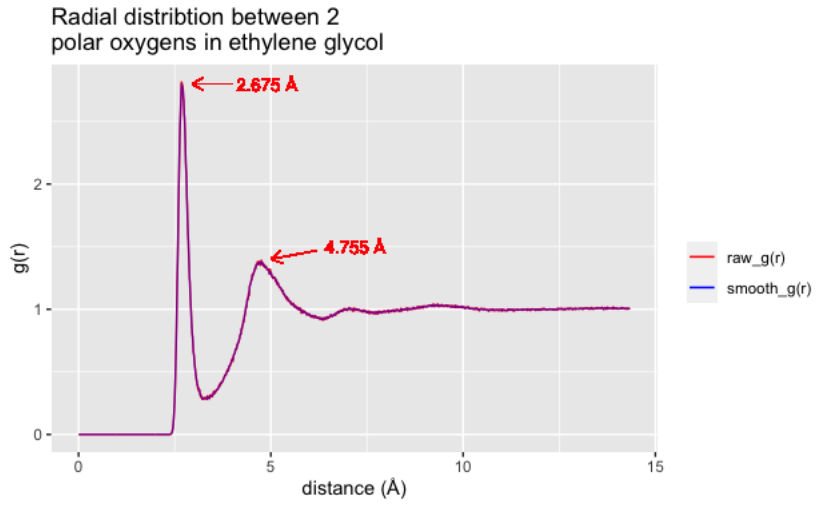
```
grep -n " 100100 Dynamics Steps" mol.EWALD.gas.log # 12018 - 1 = 12017
# remove line 18 to 12017
(head -n17 mol.EWALD.gas.log; tail -n+12018 mol.EWALD.gas.log) >
mol.EWALD.gas.equilibrated.log
mdavg mol.EWALD.gas.equilibrated.log
Total MD Blocks                4000 Blocks
 Total Energy                19.9661 Kcal/mole    (+/-   3.8336)
 Potential Energy            10.2113 Kcal/mole    (+/-   1.9599)
 Kinetic Energy               9.7547 Kcal/mole    (+/-   2.0387)
 Temperature                467.50 Kelvin        (+/-    97.70)
 Pressure                    -0.28 Atomsphere     (+/-     3.00)
 Density                     0.0039 Grams/cc      (+/-   0.0000)
```

(10.2113 - (-2896.2594/291) + 0.5921) * 4.184 = **86.84386 Kj/mol deltaHvap**
Darn

(13) Radial distribution functions can be computed from your saved MD coordinates (.arc trajectory file) using the Tinker **radial** program. For example, try computing the distribution function between pairs of like polar atoms, if your liquid contains a polar functional group. The theory and form of radial distributions is described in section 6.2.5 of Leach, and will be covered in a later course lecture. Provide a plot of the radial distribution function with your lab report.

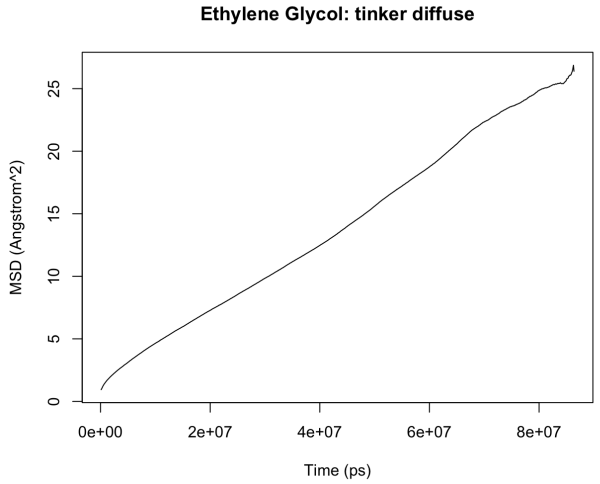
I have 2 polar oxygens
atom 108 5 OH "Diol -OH" 8 15.999 2

```
printf "1 800 1\n108 108\n0.01\nN" > radial.in
radial glycol.minimized.box_minimized.EWALD.equilabrated.arc < radial.in >
radial.out
tail -n+40 radial.out | awk '{print $3,$4,$5}' OFS='\t' > radial.graph.txt
```



(14) Use the Tinker **diffuse** program to compute the diffusion constant for individual molecules within your sample of liquid. Provide your best estimate of the diffusion constant of your liquid in your lab report, and compare to a values reported in the literature, if you can find one.

```
printf "1 800 1\n1.0\n\n291" | diffuse
glycol.minimized.box_minimized.EWALD.equilabrated.arc > diffuse.log

tail -n+39 diffuse.log | awk '{print $1,$5}' > diffuse.txt
```



(15) Finally, if your liquid can exhibit alternative conformations, such as the “chair” and “boat” conformations of cyclohexane, determine the relative percentage of each conformation as a function of the simulation time. Does it look like your system has reached its conformational equilibrium state?

Questions

(1) Why is it necessary to minimize your liquid box before starting the MD run? What would likely happen if you skipped this step?

Without performing minimization there will an increase in highly repulsive forces and maybe even attractive forces due to in compatible geometries of the molecules and clashes between the atoms of different molecules. This will cause the momentum of velocity of the atoms/molecules that is mostly likely never experienced in nature. Also, most likely we would have to increase the amount time for the equilibration stages (maybe a lot of time, even more than the actual MD step??)

(2) Diffusion constants are typically computed via the Einstein relationship. What is this relationship, and how is it used to derive the diffusion constant?

$$D = \mu k_{\mathrm{B}} T,$$

where

D is the diffusion coefficient;
μ is the "mobility", or the ratio of the particle's terminal drift velocity to an applied force, $\mu = v_d/F$;
k_B is Boltzmann's constant;
T is the absolute temperature.

gromacs

`gmx msd` computes the mean square displacement (MSD) of atoms from a set of initial positions. This provides an easy way to compute the diffusion constant using the Einstein relation.
The diffusion constant is calculated by least squares fitting a straight line (D*t + c) through the MSD(t) from `-beginfit` to `-endfit` (note that t is time from the reference positions, **not simulation time**)

The single particle velocity auto-correlation (VAC) function : Cvv (t) = < v (t) · v (0) >
The average in the velocity auto-correlation function is typically taken over **all** particles in the system and for a number of different time origins.

$D = 1/3 \int_0^\infty < v(t) \cdot v(0) > dt = uKbT$
$v(t) \cdot v(0)$ units are v^2 or [∑(dr/dt)^2]/Total molecules
https://www.reddit.com/r/learnmath/comments/omvaq/integrating_the_square_of_velocity_with_respect/
$v^2\, dt = (\Delta r/\Delta t)^2 * dt$ where Δr is just a constant w.r.t time so the integral equals:
$1/3\, \Delta r^2 \int_0^\infty 1/dt^2\, dt = 1/3\, \Delta r^2 \int_0^\infty 1/dt$

$\frac{1}{3} * \Delta r^2 * t$

$$\int_0^\infty < \mathrm{v}\,(\mathrm{t}) \cdot \mathrm{v}\,(0) > \,\mathrm{dt} =$$

$$\int_0^\infty \frac{\sum(\Delta r/\Delta t)^2}{total\ mols}\,\mathrm{dt} =$$

$$\int_0^\infty \text{mean of } (\Delta r/\Delta t)^2\,\mathrm{dt} =$$

- mean of $(\Delta r/\Delta t)^2$ *is just units of* $(\Delta r/\Delta t)^2$

$$units\ of \rightarrow \Delta r^2 \int_0^\infty \left(\frac{1}{\Delta t}\right)^2 \mathrm{dt} =$$

$$units\ of \rightarrow \Delta r^2 \int_0^\infty \left(\frac{1}{\mathrm{dt}}\right)^2 \mathrm{dt} =$$

$$units\ of \rightarrow \Delta r^2 \int_0^\infty \frac{1}{\mathrm{dt}}$$

The Δr^2 makes sense as this is the MSD but how does integrating $\mathbf{(1/\Delta t)^2 dt}$ give units of 1/time? (Nick helped out explaining this to me)

Einstein relation
u: units are $\mu = v_d/F$; m/s / J
Kb: units are The Boltzmann constant has SI units of $\mathbf{J{\cdot}K^{-1}}$
T: units are K
Final units are: m/s /J * J * K^-1 *K = m/s divided by 6 degrees of freedom

The slope of is the line is also in MSD/time. **D** measures mobility (transport) of atom/molecule through time based on relative positions. $\vec{\boldsymbol{D}}$ is still a vector?? That is, the average of the squared distances differentiated over the total time to reach those relative distance changes.

(3) What methods can be used to determine whether an MD simulation is "equilibrated"? Not all properties equilibrate and/or convergence at the same rate. Which kinds of properties tend to converge more slowly?

The two I know of so far are NVT and NPT. There are more, sounds like the constant energy can replace constant temperature in NVT -> NVE. NVT is looking for temperature convergence and NPT is looking for density convergence????

From Justin on NVT:
"This ensemble is also referred to as "isothermal-isochoric" or "canonical." The timeframe for such a procedure is dependent upon the contents of the system, but in NVT, the temperature of the system should reach a plateau at the desired value. If the temperature has not yet stabilized, additional time will be required."

Another note from Ross Walker maybe due to the fact that NVT convergence may depend more on the input parameters such as heating the system.

Example simulation. Lysozyme in water tutorial: Same .mdp parameter for the most part except Pressure coupling is turned on. Ran with 1 MPI thread and 12 OMP threads on MacBook Pro.

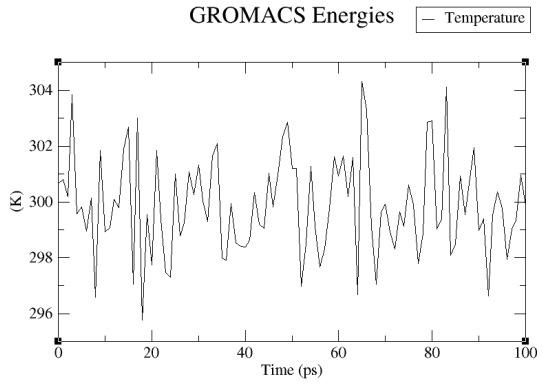
```
; Run parameters
integrator              = md        ; leap-frog integrator
nsteps                  = 50000     ; 2 * 50000 = 100 ps
dt                      = 0.002     ; 2 fs
```

NVT (100 ps)

```
               Core t (s)   Wall t (s)        (%)
       Time:     3745.907      312.163     1200.0
                 (ns/day)    (hour/ns)
Performance:       27.678        0.867
```

Plot Temperature convergence

```
printf "16\n0\n" | gmx energy -f nvt.edr -o temperature.xvg | xmgrace temperature.xvg
```

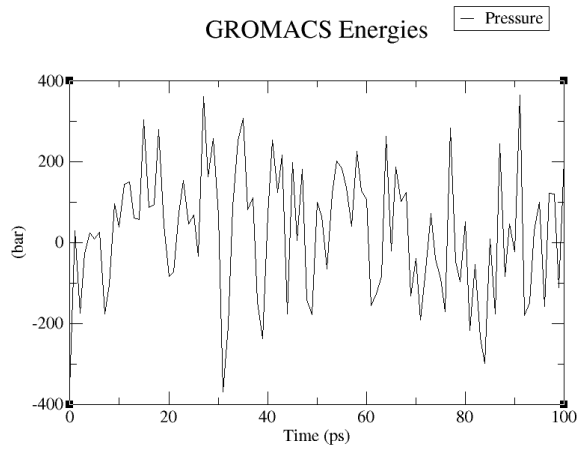


NPT (100 ps) similar wall time

```
               Core t (s)   Wall t (s)        (%)
       Time:     3559.303      296.612     1200.0
                 (ns/day)    (hour/ns)
Performance:       29.130        0.824
```

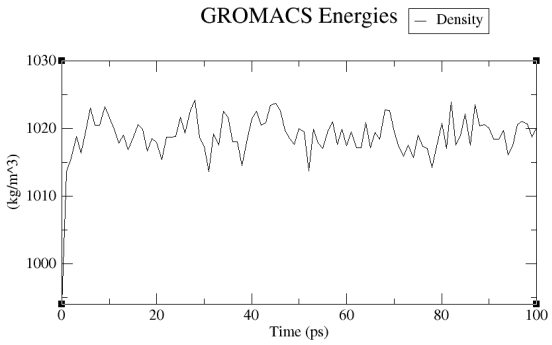
Plot pressure and density convergence:

```
printf "18\n0\n" | gmx energy -f npt.edr -o pressure.xvg | xmgrace pressure.xvg
```




```
printf "24\n0\n" | gmx energy -f npt.edr -o density.xvg | xmgrace density.xvg
```

Looks as if density convergence seems less erratic.



(4) It is also possible to use an MD simulation to compute an estimate of the heat capacity of your liquid. Find the statistical mechanical fluctuation formula for the heat capacity and apply it to your simulation data (see Leach or a similar book). The value you derive will probably be rather uncertain, at best. Why?

Mine was pretty far off. Simulated deltaHvap = 86.84386 kJ/mole. The experimental value (average of 13 studies) was only 65. ± 3. kJ/mol. In order to perform the calculation, I did a simulation of 1 ethylene glycol molecule at its BP temperature of 470K and ran similar MD step and removed equilibration period. I am postulating a couple reasons for extreme difference:

1) The (+/-) temperature change from mdavg was really high for the single molecule gas phase (+/- 97.70 K) vs liquid phase (+/- 3.25 K). This would cause inacurracies I

think. Should the temperature be fluctuating? Do we need to constraint the thermostat with keywords???

2) Just thinking from a high-level perspective outside of MD. If we want to actually do this experimentally, would we really want to only learn the properties of the system of single gas molecule? I think we'd actually want to increase the temperature over time from melting point to boiling point like in the protocols ("The temperature of the sample in the boiling flask (distilled water) is increased until boiling commences"). Then we extrapolate the properties of the system as an average of the properties as every molecule transitions from liquid phase to gas phase.
3) The Clausius-Claypeyron equation does an approximation outside of a physical experiment also. Will look into this.

The experimental dipole moment for **water** is **1.8 Debye**