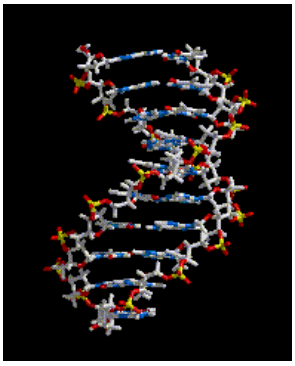
(Rev 9/8/08)

Amber: Building DNAs and Minimizing their Energy
Lab 4

Table of Contents

Objectives



Pictured at left is an average structure taken over 1 nanosecond from a solvated particle mesh Ewald simulation of a decamer poly(A)-poly(T) duplex. The average structure was created using `carnal` to RMS fit and coordinate average all DNA atoms from the trajectory taken at 1 picosecond intervals.

Purpose

The purpose of this lab is to demonstrate how to set up a standard decamer poly(A)-poly(T) duplex DNA model structure using the tools provided with AMBER 8 and more generally provide information on setting up and running AMBER for DNAs and Proteins for molecular mechanics and molecular dynamics.

Outline

This lab is divided as follows:

Using **nukit** and **nucgen** to build canonical DNA structures.
What should I simulate? A discussion of issues to think about before starting.
Fixing the initial model with **sander**.
Building the `prmtop` and `inpcrd` files and what they are.
Using **xleap** to build.

After we've built up the *in vacuo* `prmtop` and `inpcrd` files we will run **sander** to perform a rough minimization. Then, we will add solvent and counter ions and build a second set of `prmtop` and `inpcrd` files for molecular mechanics energy minimization. Eventually we will get to the point where we can create the picture shown above. In subsequent labs we will include a description of one possible way to "equilibrate" the system and then move into a brief discussion of performing "production" molecular dynamics runs. After initial equilibration, all of the simulations will be run using the **particle mesh Ewald** (PME) method [J. Chem. Phys. **103**, 8577-8593. (1995)] within AMBER 8. Some trajectory analysis issues with the current version of the PME code will also be discussed.

Definitions

In this discussion, we first figure out how to generate a starting structure and then use this structure to build up the input files necessary for **sander**. The basic files necessary to run **sander**, the program used in Amber for molecular mechanics and molecular dynamics calculations, are:

`prmtop`: a description of the molecular topology including definition of atom type, bond lengths, angles, etc., and force field parameters. In some labs topology files will have the extension `.prmtop` and in others it will be `.top`.

`inpcrd`: (or a `restrt` file from a previous run) a description of the coordinates and optionally the velocities and current box dimensions. In some labs topology files will have the extension `.inpcrd` and in others it will be `.crd` (see also `restart` files).

`mdin`: the **sander** input file which is a series of namelists and control variables. The typical file extension will be `.in`.

`mdout:` the **sander** output file which has the file list used for the run, a copy of the input file, and the energy calculation output. Often if a run fails immediately, this file will contain diagnostic messages (which may or may not be useful). The usual file extension is `.out`.

`restrt:` this a **sander** output file that contains the coordinates of the final geometry. When combined with the `prmtop` file using **pdbgen**, a pdb file containing the final geometry is produced. The file extension used is typically either `.restrt` or `.rst`.

The combination of `prmtop` and `inpcrd` define the structure of whatever you are trying to model. These files are required input into **sander** and can be used to generate a pdb file. The `mdin` file is the file that controls how **sander** runs. Some of this file will be discussed here as it pertains to what calculation we are doing.

Setup

Preliminaries

First login onto a unix machine and change to the previously created (Lab 2) ~/dna_data

directory. If you did not create this directory then after you log on, enter the command:

```
mkdir dna_data
cd dna_data
```

You also need to copy a few files to this directory. They are located in /usr/people/amberlab4. Issue the command shown below to copy the necessary files to your ~/dna_data directory:

```
cp /disk02/usr/people/amberlab/amberlab4/* .
```

The following files should be copied:

`fixit.in`	This is a `mdin` file. A copy of it is listed in Appendix A.
`nucgen.dat`	Data file needed for the program **nucgen**.
`polyAT_fsi.in`	This is another `mdin` file.

Building the initial canonical DNA structure

Introduction

The first step in starting any modeling project is developing the initial model structure. Typically, one searches structural databases of crystal or NMR structures. With nucleic acids, users can also search the Nucleic Acid Database (http://ndbserver.rutgers.edu/) maintained by Helen Berman at Rutgers.

When working with nucleic acids, in the absence of a good experimental structure, all hope is not lost since there are also a variety of programs out there that facilitate building model nucleic acids structures; the list of possible sources is beyond the scope of this lab. However, it is worth mentioning that Dave Case and Tom Macke of Scripps have developed the **NAB** molecular manipulation language (http://casegroup.rutgers.edu/casegr-sh-3.2.html) which facilitates the building of complex nucleic acid structures. Alternatives to **NAB** include **Sybyl** (installed on Iris), **Insight** (installed on Frodo) and **McSym** (http://www-lbit.iro.umontreal.ca/mcsym/). **NAB** and **McSym** are installed on Frodo.

---

nukit and nucgen

Overview

Amber also comes with some programs which facilitate building standard canonical A- and B-duplex geometries of nucleic acids and this section will demonstrate there use.

The program **nucgen** which comes with the AMBER 8 release will build Cartesian coordinate canonical A- and B- models of standard DNA:DNA, RNA:RNA or DNA:RNA duplexes. Input files for **nucgen** can be prepared using the program **nukit** or, if you know what you are doing, with a text editor. The `pdb` that is output will have all the atoms named according to the AMBER naming conventions; many of the hydrogen atoms will be missing and hence will need to be added. **xleap** or **edit** conveniently does this. In this lab we will use **xleap**.

The **Amber** force field, the Cornell et al. (1995) or `parm94.dat` force field, was parameterized with special attention towards nucleic acids. For simulations in explicit solvent, this force field is recommended over earlier force fields. To use this force field, distinctions need to be made in the terminal residues; so therefore we now have names such as A5, A, and A3 for a 5' terminal adenine, a non terminal adenine, or 3' terminal adenine respectively (Note, the abbreviations actually are prefaced with either a D or an R depending on whether you are working with DNA or RNA).. To build a DNA or RNA for use, here, you will use the programs **nucgen** and **nukit**.

nukit

Here is a sample session for building a decamer poly(A)-poly(T) DNA duplex in the Arnott canonical B geometry with `nukit` and the `parm94.dat` naming conventions. To start **nukit** simply type the command shown. (What is shown following this command below is the computers response and what your input (bolded) should be.)

```
nukit

Residue naming convention? (O = pre-94, N = 94) O/N:
N
JOB NAME?
poly A poly T
--------(from here on, USE CAPITALS) ---------
--------- e.g. CGCGATAT ---------

 ENTER SEQUENCE {5-prime to 3-prime}:
AAAAAAAAAA
 DNA OR RNA? (D/R):
D

 ENTER SEQUENCE {5-prime to 3-prime}:
TTTTTTTTTT
 DNA OR RNA? (D/R):
D
             CONFORMATIONS:
     ARNA   right handed A rna (arnott)
     APRNA  right handed A-prime rna (arnott)
     LBDNA  right handed B dna (langridge)
     ABDNA  right handed B dna (arnott)
     SBDNA  left handed  B dna (sasisekharan)
     ADNA   right handed A dna (arnott)
     NIXON  none of above - nucgen.pdb user-supplied

  A-forms may need work to place H1-primes properly.
```

```
 CONFORMATION?
ABDNA
```

After the last response the program will terminate and the files `nuc.in` and `lin.in` will be generated. Note that if you need to re-run nucgen, delete nuc.in and lin.in first.

The program **nukit** writes the file `nuc.in`. The file `nuc.in` should look exactly as shown below (Note, the first line of the file begins with NUC 1, not a blank line):

```
 NUC  1
D
A5  A   A   A   A   A   A   A   A   A3

 NUC  2
D
T5  T   T   T   T   T   T   T   T   T3

END
$ABDNA
```

nucgen

To run **nucgen**, you need to specify the input file, an output file, the **nucgen** database and the name of the pdb you wish to write. The **nucgen** data base (`nucgen.dat`) should have been copied at the beginning of this lab. If not, copy it now from /usr/people/amberlab4. The command to run **nucgen** is shown below. You will notice that just after **nucgen** is -O. This shows up for many Amber programs. What it means is that any output files that exist in the current directory will be overwritten (and there are two such files, `nuc.out` and `nuc.pdb`).

```
nucgen -O -i nuc.in -o nuc.out -d nucgen.dat -p nuc.pdb
```

Now that you've got a model pdb, you may want to look at it. You can use **moilview** or **rasmol** to do this. But first, the file `nuc.pdb` has to be slightly modified. In order for **xleap** to "figure out" where one strand terminates and the next begins, it is necessary to add pdb TER cards to the `nuc.pdb` between the strands since **nucgen** unfortunately doesn't do this. The modified file will be called `nuc_ter.pdb`. To make this file, edit `nuc.pdb` in jot and then save it as `nuc_ter.pdb`. To begin enter:

```
jot nuc.pdb
```

and scroll through the file until you find the line separating base 10 from base 11. This should be somewhere around line 229. After the last line for base 10, insert a line with

```
TER
```

Then pick ‘File/Save as’ and save the file as `nuc_ter.pdb`. If you want to be really sure that **xleap** won’t have problems reading this file, go to the end of the file and insert a TER there, too.

You should also make sure there is one blank line at the end of this file.

---

rasmol

You can view `nuc_ter.pdb` with **rasmol** just to make sure it looks O.K. A really convenient way to run **rasmol** and load the file of interest at the same time is to enter the command:

```
rasmol nuc_ter.pdb
```

Note that if you are on taz or kali (running linux) you can view the pdb file by running pymol. On the command line just type pymol and load the file through the file menu or, alternatively, just enter

```
pymol nuc_ter.pdb
```

---

moilview

Loading molecules into **moilview**

Moilview is not without its bugs, it only runs on the unix machines, and it does not run in a useful way remotely, but is still a very useful program for viewing `pdb` files and other files generated from molecular dynamics. A major draw back is that it will not run over the network nor is there a Windows version of this program. So, if you are on any of the SGIs moilview will work but does have to be run locally. If you are on a PC, you will have do this part of the lab at a later time. To run the program, type the following command:

```
moilview
```

After **moilview** starts right click in the main window and from the pull-down menu select "Read 1st coordinates" to load the pdb file. At the prompt, select pdb and from the dialog box choose the file you are interested in viewing (e.g. nuc_ter.pdb). When prompted again, select "use DISTANCE bonds". This will infer bond based on distance. If you have AMBER `prmtop` and `inpcrd` files (and right now you don't) you may choose this to specify the connectivity. Note that after you have chosen the file format once (i.e. pdb vs. coordinate, etc), you need not choose it again unless you want to change file formats. Therefore, if you want to load something with a different format, you may need to change the format from the right button "file" menu and "set file formats" submenu.

Examine the structure of the DNA you have built. Note that the perfect symmetry of canonical duplexes is based on analysis of long fibers of DNA. Real nucleic acids don't necessarily adopt this perfect symmetry as will be apparent in the dynamics; this is particularly true with poor

force fields, lack of solvation, and not properly treating the long ranged electrostatic interactions. Developing the simulation methods such that we can accurately represent the structure and dynamics, and moreover sequence dependent nucleic acid fine structure, is a current problem of much research interest.

Preparing a plot

At the end of this lab you will be asked to turn in pictures of your work. It might be most convenient to produce your plot now. You could print this structure as shown but it is a little rough and can be improved and, at the same time, you will learn more about **moilview**. So, what we you should do is:

1) Switch the background from black to white (and this saves toner so please do it). This is accomplished by right-clicking in the view window, select 'color options/background color/white black.' After you have done this the background should switch to white.
2) Switch the presentation to ball-and-stick or space-filling (your choice). This is accomplished by right-clicking in the view window, select 'Objects are/Ball and cylinder (or spheres). The displayed molecule should be a ball and stick (space filling) model. If you want to change the size of the spheres right-click in the view window, select 'Change parameters/Ball and Stick Params/Sphere Scale Factor' (something between .15 and .25 works best).
3) Produce a post-script file. You must first produce a post-script file and then print from a Unix shell. You can't print directly from **moilview**. The post script file is produced by right-clicking in the view window, selecting 'Plots/Postscript drawing/Ball and Stick.' You will be asked for a file name - respond with nuc_ter_pdb.ps.
4) Print the post-scipt file. To print the file either exit **moilview** (right-click in the view window, selecting 'Quit') or open another Unix shell. The file you are looking for is in the directory that you were running **moilview** from. Change to that directory and then enter the following command:

```
lp nuc_ter_pdb.ps
```

That's it. The data light on the printer (rm 2025) should start blinking and shortly thereafter, your plot will be spooled.

What level of simulation am I going to attempt?

You may want to skip to the next section and read the following after you have completed the lab. It is not essential for the lab, but the discussion is useful in regard to modeling of nucleic acids in general.

After you've got your model structure, it is time to decide what level of simulation realism you

are willing to bring to bear on your problem. The complexity of the calculation centers on the evaluation of the pairwise nonbonded and Coulombic interactions. Extra complexity comes in from using Ewald methods to treat the long ranged electrostatics or evaluating non-additive effects.

Water is an integral part of nucleic acid structure, so some representation of solvent is fairly critical. Simulations *in vacuo* have been performed where the screening of solvent is modeled by distance dependent or sigmoidal dielectric functions (the latter isn't implemented in AMBER 5.0). Additionally, tricks have been applied to keep the base pairs from fraying (through the addition of Watson-Crick base pair restraints) and charges on the phosphate reduced.

If you plan to run *in vacuo* simulations with nucleic acids, rather than the standard Na+ (IP) counterion, it is probably wiser to use a softer and larger counterion that effectively represents the ion plus its integral water shell [Singh, H., Weiner, S.J. & Kollman, P.A. *Proc. Nat. Acad. Sci.* **82**, 755-759 (1985).]

A number of simulations have been run on DNA with explicit solvent. Earlier treatments employing molecular dynamics simulations with explicit solvent [for reviews, see Beveridge and Ravishankar, *Cur. Op. Struct. Biol.* **4**, 246-255 (1994) or Louis-May, S., Auffinger, P. & Westhof, E. *Cur. Op. Struct. Biol.* **6**, 289-298 (1996) and references therein] were limited to a short time scale (~100 ps) and typically displayed anomalous structure (such as base pair fraying). These simulations demonstrated the need for inclusion of a more accurate representation of solvent. More recent simulations on a longer time scale (~1 ns) suggest the importance of properly treating the long ranged electrostatic interactions [see, for example, Cheatham et al. *J. Amer. Chem. Soc.* **117**, 4193-4194 (1995), Cheatham & Kollman, *J. Mol. Biol.* **259**, 434-444 (1996) and the references cited above].

However, even with advances in computer power and methodological improvements, such as application of Ewald methods, which allow routine simulations of nucleic acids with explicit solvent and counterions in the nanosecond time range, there is still dependence of the results on the molecular mechanical force field. For example, Yang & Pettitt observed a B-DNA to A-DNA transition [Yang & Pettitt, *J. Phys. Chem.* **100**, 2564-2566 (1996)] when the CHARMM-23 all hydrogen parameter set [Version 6.1, November 1993] was applied with an Ewald treatment on the dodecamer d[CGCGAATTCGCG]$_2$ which suggests that the A form of this structure is more stable. On the other hand, we observe that B-DNA is more stable than A-DNA when the force field described by Cornell et al. is applied in molecular dynamics simulations with the particle mesh Ewald method within AMBER 5.0.

Now back to the point of this discussion, if you can afford it, include the solvent. Definitely include the explicit net-neutralizing counterions. Also pay attention to the force field applied and its limitations. Use Ewald methods if you can. However remember that adding water is expensive. While I can get on the order of nanosecond or so of DNA simulation in less than a day on an SGI R10000 *in vacuo*, adding a periodic box of water that surrounds the DNA by roughly 10 angstroms extends the simulation to ~17 days. Given that you generally need to run the simulation a couple of times (due to errors, sampling issues, etc), these simulations get costly.

For this series of demos, we will build an *in vacuo* model of the poly(A)-poly(T) structure (named polyAT_vac) and a TIP3P (water) solvated model of the poly(A)-poly(T) structure in a periodic box (polyAT). This solvated model can then be applied in standard cutoff simulations or with a particle mesh Ewald treatment.

In order to simplify analysis of the trajectories later, it is very useful to have both sets of `prmtop` files. This is because often in the analysis of the trajectory solvent isn't necessary. Obviously the water is necessary for calculating radial distribution functions, analyzing water structure, and other properties, however it isn't necessary for calculating helicoidal parameters, determining average structures, etc. Therefore, to minimize disk space usage, and speed the analysis, we often strip the water and/or counterions. The two separate `prmtop` files are useful to have around since you want to use a `prmtop` that matches your structure or (possibly stripped) trajectory for programs such as carnal, **moilview**, **MD Display**, **rdparm**, **pdbgen**, etc.

Building the `prmtop` and `inpcrd files`

Now that we have a starting pdb (`nuc_ter.pdb`) we are ready to build the input files necessary for **sander**. This will be done using **xleap**.

As long as you are using standard nucleic acid or amino acid residues, you can use the residues prebuilt into the AMBER database. These are stored in two forms, the format suitable for `prep` and the premanaged library suitable for **xleap**. Fortunately, **xleap** can also accept the `prep` format (with the `loadAmberPrep` command). Since we are using standard adenine and thymine nucleotides, we can avoid this stage altogether.

Outline

In this section you will:

1) Start **xleap** and load the nucleic acids database (`all_nuc94.in`)
2) Load the pdb file `nuc_ter.pdb` simultaneously adding the missing hydrogen atoms and inspect the structure
3) Write the corresponding `prmtop` and `inpcrd` files
4) Run **Sander** to fix up the structure caused by **xleap** adding hydrogens
5) Solvate your DNA
6) Add counter ions to your structure
7) Write the new `prmtop` and `inpcrd` files
8) Run **sander** to minimize the structure.

---

xleap

Start up **xleap**:

```
xleap
```

After you start **xleap** you get a bunch of messages indicating what has been loaded. In so doing, you should have seen that the appropriate libraries were loaded.

You can check to see if the residues have been properly loaded by trying to "edit DA5". If the unit isn't blank, the residues were properly loaded.

If you do get a blank screen when you try to edit DA5, you can enter the following commands after you enter **xleap** (lines beginning with > have the commands you should enter. Lines with out the > give the response to the command. Note that you will have to determine what $AMBERHOME is and replace it in the command line below, since as noted above, **xleap** does not understand $AMBERHOME.

```
> loadAmberPrep "$AMBERHOME/dat/leap/prep/all_nuc94.in"
```

This bring's up the issue of how thing's are initially set up. If you were to play around with **xleap** and edit the unit named "A", up would pop alanine; likewise, "T" is threonine. This seems to conflict with the DNA names. Here are the different names of the residues for the O3' atoms from the pdb we created above.

```
ATOM      2  O3' A5      1       4.189  -7.682  -0.250
ATOM     25  O3' A       2       7.904  -3.753   3.130
ATOM    209  O3' A3     10      -1.127  -8.677  30.170
ATOM    231  O3' T5     11       7.904   3.753  30.670
ATOM    252  O3' T      12       8.601  -1.610  27.290
ATOM    420  O3' T3     20       4.189   7.682   0.250
```

How does **xleap** recognize that the "A" there is adenine and not alanine? How does **xleap** recognize that this is DNA and not RNA? Equivalently, how does **xleap** know to map A5 to DA5 and A to DA instead of ALA or RA5 or RA? This has to do with a startup file called the "`leaprc`" file that resides in the **xleap** (`/$AMBERHOME/dat/leap/cmd`) directory. If you look through this file, you will notice the command:

```
#
#  A map for nucleic acids - assuming always DNA
#
addPdbResMap {
  { 0 "C5" "DC5" }
  { 0 "G5" "DG5" }
  { 0 "A5" "DA5" }
  { 0 "T5" "DT5" }
  {   "C"  "DC"  }
  {   "G"  "DG"  }
  {   "A"  "DA"  }
```

```
  {   "T"  "DT"  }
  { 1 "C3" "DC3" }
  { 1 "G3" "DG3" }
  { 1 "A3" "DA3" }
  { 1 "T3" "DT3" }
}
```

This maps the names of pdb residues to **xleap** residues. We see that by default we assume everything is DNA. In addition to the above mapping, we see that there are mapping of `pdb` style naming conventions to AMBER, such as `O4*` to `O4'`, etc. This means when using **xleap** and non-standard residues molecules, such as DNA:RNA hybrids, you cannot simply expect to use the `loadPdb` command and get reasonable behavior. In this case you may move to the more advanced command, `loadPdbUsingSeq`, which allows overridding the `pdb` residue names and give you much more control.

Now, let's go back to the beginning and assume everything was set up properly; the distraction above was just to give a little insight behind the scenes. We are now ready to load up the pdb file we previously created.

Now load up the pdb into **xleap** with the loadpdb command and then edit it with the edit command (remember to substitute pgannet2 with your userid).

```
> model = loadPDB nuc_ter.pdb
Loading PDB file: /disk02/usr/people/pgannet2/dna_data/nuc_ter.pdb
  total atoms in file: 438
  Leap added 200 missing atoms according to residue templates:
> edit model
```

You can also load this file using the pull down menus. Click on File, Load PDB. Navigate to the sub-directory `./dna_data` and highlight `nuc_ter.pdb`. In the information box labeled Variable, enter `model` and then click on accept.

Fixup of Hydrogens

Using **xleap** (or **edit**), we allowed the programs to add the missing hydrogen atoms (note the message after you load the pdb file that says Leap added 200 missing atoms). As mentioned in the discussion on building residues, the positions of the "to be placed" hydrogens depends on the geometry specified in the residue database. This geometry may not correspond to the geometry in the actual molecule and certainly doesn't consider possible "conflicts" with other residues. This problem is particularly acute when adding hydrogens to terminal groups with edit; sometimes edit cannot place these properly and they are placed way out in space.

Therefore, whenever hydrogens are added with **edit** or **xleap** (and probably also with **protonate**, another Amber program), and before continuing on to add counterions and water, first "fix up" the positions of the hydrogens. This "fixed" pdb is what will be use as the reference pdb for all root-mean-square coordinate deviation measurements, the initial structure of all the coordinate

inputs, etc.

When we fix up the hydrogen positions, we do not want to alter the heavy atom positions (particularly if we are using an experimental structure). A simple way to fix the hydrogens is to go through the procedure below to create a `prmtop` and `inpcrd` suitable for an *in vacuo* run, and then minimize the positions of the hydrogens with **sander** using the 'belly' option. Appendix A has a copy of the `fixit.in` file and some discussion about this file.

To create the necessary **sander** input files, exit the unit editor and type:

```
> saveamberparm model polyAT_vac.top polyAT_vac.crd
> savepdb model polyAT_vac.pdb
```

Open another shell and change directories to the `~/dna_data` directory. Check and make sure you have the following files:

```
fixit.in
polyAT_vac.top
polyAt_vac.crd
```

One important thing to check in belly simulations is that only the atoms you want to move are actually able to move, i.e. be careful to check the list of atoms in the belly. For more information on the GROUP option for inputting the list of atom names, types, etc, see the appendix in the **Amber** 7.0 manual. Here is a **sander** run to fix the positions. Enter the following command (don't include the '\', they are there to indicate that the command is one continuous line, no carriage returns until after the '&')::

```
sander -O -i fixit.in -o fixit.out -p polyAT_vac.top \
       -c polyAT_vac.crd -r polyAT_vac.rst \
       -ref polyAT_vac.crd &
```

The **sander** run should only take a few seconds (about 34) to complete. When it is done, generate a pdb file using **ambpdb** and the files `polyAT_vac.top` and `polyAT_vac.rst` (this file has the new coordinates in it). It would be wise, also, to copy `polyAT_vac.top` to `polyAT_vac_fixed.top`, move `polyAT_vac.rst` to `polyAT_vac_fixed.crd` (use the mv command). Name the output file from **ambpdb** `polyAT_vac_fixed.pdb`. Finally, be sure to check to make sure there is TER line between the two strands and, to be safe, at the end of the strands. They should be inserted between (approximately) line 319 and 320 and after line 638 (639 if you include the TER line, just inserted). Again, this can be done with **jot** or **nedit**. On linux boxes use **gedit** or **nano**.

In the Questions section you will be asked to prepare a figure of `polyAT_vac.pdb` and `polyAT_vac_fixed.pdb`. It is probably better to generate these plots now. Refer to the section on '**moilview**/Preparing a Plot' section to guide you through this process.

To make a pdb from a coordinate (or restart, here we are using the restart file) and topology file

type:

```
sander -p polyAT_vac.top < polyAT_vac.rst > polyAT_vac.pdb
```

Solvating and adding counterions

Now, return to **xleap** and load `polyAT_vac_fixed.pdb` (unit = model) and look at the structure in the unit editor. (You may also want to get the starting structure and fixed structure in **moilview** since you can compare them in this program.

If the model (`polyAT_vac_fixed.pdb`) looks O.K. then exit the unit editor (File/close). The next input file to create is the solvated DNA with explicit counterions. We have our "model" unit already built. Now we need to solvate it. This is done with the command "solvateBox". For our DNA, we will put ~10 angstroms of water around the DNA in each direction.

```
> solvateBox model TIP3PBOX 10.0
```

Now we want to add explicit net neutralizing counterions. Here, we are going to use a method built-in to **xleap**. This method works by constructing a Coulombic potential on a 1.0 angstrom grid and places the counterions one at a time at the points of lowest/highest electrostatic potential. The command is as follows (the '0' means 'neutralize'):

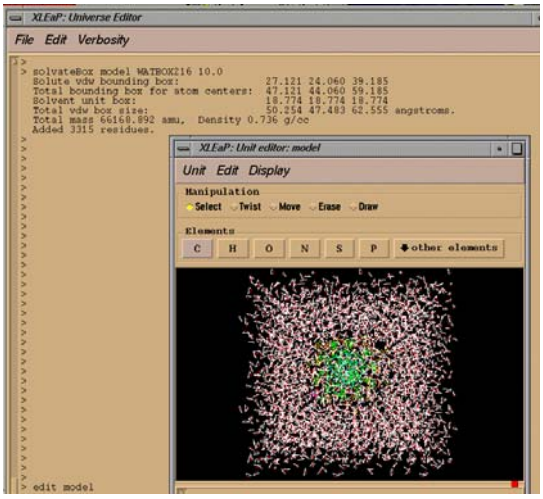
```
> addIons model Na+ 0
```

(NOTE: The character after Na+ is a zero not an uppercase ‘o’.)

Of course you should carefully check the output from the counterion placement to determine if as many ions were placed as you intended. To check your model now, type

```
> edit model
```

Your screen should look something like the picture to the right.



If you see something that looks like the picture above, you can see that the box size came out to be ~50 by 47 by 62.5 angstroms[3]; this is not cubic since the DNA is a cylindrical molecule. An issue here is that the long axis of the DNA could rotate (via self diffusion) such that the long axis was along the short box dimensions which will bring the ends of the

DNA near their periodic images. You can get around this problem by actually making the box cubic, or ~62.5 by 62.5 by 62.5 angstroms^3, by specifying a list of numbers to the solvateBox command to force this to be cubic. However this will add significantly more water to the calculation and slow it down tremendously.

Note that if you are going to play around with the solvate command, you might want to make a copy of your **xleap** unit before solvating. To copy a unit, you can use the command `copy` within **xleap**.

```
> new_model = copy model
```

Also, it is easier to equilibrate if we add water, then ions, since in this case addIons will replace the closest overlapping water with the ion, which gives better packing. However, the 'ions-first' approach, not followed here, allows us to save a topology file with only DNA and ions for later use in analysis of trajectory files that have had the waters stripped to save disk space. A work around will be described. However, you might want to prepare `top` and `crd` files corresponding to a system to which the ions are added first and then water. If so, replace `fsi` with `fis` in the file names (just to keep things straight) and reverse the steps described above (ie use addIons command first, then the solvate command). Run both systems and compare the energies to see if similar results are obtained. You can also remove the waters from each system and use **moilview** to determine the RMSd between the two systems.

Now once again we are ready to write the `prmtop`/`inpcrd` and `pdb` files. Exit the unit editor and type:

```
> saveAmberParm model polyAT_fsi.top polyAT_fsi.crd
> savepdb model polyAT_fsi.pdb
```

In the Questions section you will be asked to prepare a plot of `polyAT_fsi.pdb`. It is probably better to generate this plot now. Refer to the section on '**moilview**/Preparing a Plot section to guide you through this process.

Equilibrating and running the poly(A)-poly(T) structure

Introduction

After you've built up your structure, you are ready to begin minimization and molecular dynamics simulations. In general, before production dynamics should be run, potentially bad van der Waal (nonbonded) and electrostatic interactions should be minimized. If you used **xleap** or edit to solvate the structure, water was placed from a pre-equilibrated water box around the solute. Therefore this water hasn't felt the influence of the solute and moreover there may be gaps between the solvent and solute and solvent and box edges. Therefore, before one runs production dynamics on the system, it is wise to let the water relax (or equilibrate) around the

solute to come to an equilibrium density, etc., and unless the water density is perfectly matched to the box size and all "holes" around the solute filled, constant pressure dynamics should be used to allow the box size to change. Otherwise (if constant volume simulations are run), "vacuum" bubbles may appear in the box. In this section is presented one path towards equilibrating a solvated DNA structure for use in simulations with the particle mesh Ewald (PME) method. After equilibration is demonstrated, we set up a production "PME" run. There are many differing opinions on how one should equilibrate a simulation. Herein is presented only one.

NOTE: In this tutorial, you will not run for an entire equilibration cycle on the input files generated in the previous tutorial (since it takes ~1 day to run on an SGI R10000). Instead, we will show the inputs and command lines necessary to run the equilibration procedure and discuss some of the choices made. After this discussion, final results will be shown that I have generated.

One big issue with the constant pressure simulations is that in order to match a target pressure, the box size is changed by scaling of the atom or molecule positions to match a target pressure [See Berendsen, H.J.C., Postma, J.P.M., van Gunsteren, W.F., DiNola, A., & Haak, J.R. *J. Comp. Phys.* **81** 3684-3690 (1984)]. This means that one has to be very careful using belly simulations with constant pressure. A belly simulation (IBELLY = 1) implies that you have chosen a subset of the system to be held fixed. This is performed by zeroing the forces on each of the fixed atoms at each step.

However, atom positions may change as a result of pressure scaling. If atom based scaling is used (NPSCAL = 0), then if all the solute atoms are held fixed, the positions of the individual atoms will be shifted as the box size changes. This means that you will move away from your initial structure, even with a belly on all the solute atoms. On the other hand, if you use molecule based pressure scaling (NPSCAL = 1), then only the center of mass of the molecules will be shifted. This is fine if your solute is only a single molecule. However, the two strands of the DNA are each in a separate molecule. Therefore, without some trickery, you would be advised to run a (slightly slower) simulation with position restraints (NTR = 1) rather than using a belly.

Step one: Minimization holding the solute fixed

The first step is a short minimization run to remove any potentially bad contacts. In this equilibration, you'll use belly; however, using position restraints is much more general and advised (unless you modify your `prmtop` file to support belly equilibration). In this step, we are placing positional restraints of 500.0 kcal on each of the solute atoms. Thing's to notice in the input file (`polyAT_fsi.in`)
:

`IMIN = 1` Minimization is turned on.
`SCEE = 1.2` Since we using the `parm94.dat` force field the 1-4 electrostatic

interactions must be scaled by 1.2. If we were using `parm91.dat`, this scale factor would be 2.0.
`IDIEL = 1, DIELC = 1.0` Use a constant dielectric constant of 1.0.
`NTR = 1` means we have turned on position restraints and therefore have to specify via GROUP input (see the appendix of the AMBER 7.0 manual) the atoms which are restrained as well as the force constant.

In this example, we use a force constant of 500 kcal/mol-angstrom and restrain residues 1 through 20. This means that the water and counterions are free to move. This GROUP input looks like the following:

```
Hold the DNA fixed
500.0
RES  1  20
END
END
```

Note that whenever you run using the GROUP option in the input file, carefully check the top of the output file to make sure you've selected as many atoms as you thought you did.

```
GROUP 1 HAS HARMONIC CONSTRAINTS   500.00000
GRP    1 RES    1 TO 20
Number of atoms in this group =   638
END
```

MAXCYC = 1000, NCYC >= MAXCYC: This means that we will be running 1000 steps of steepest descent minimization. Conjugate gradient minimization typically fails with rigid water (TIP3P) so by keeping NCYC > MAXCYC we never turn it on. This avoids the common "LINMIN failure".

NTC = 1, NTF = 1: When running dynamics with a rigid water, shake must be turned on. However, it is not necessary with minimization.
CUT = 9.0, NSNB = 10: We are running with a 9 angstrom cutoff and updating the pairlist every 10 steps.

When running NTR = 1 simulations, the reference coordinate file for the restraints must also be listed. It is the same format as the standard **amber** coordinates and must match the `prmtop`.

```
sander -O -i polyAT_fsi.in -o polyAT_fsi.out /
-c polyAT_fsi.crd -p polyAT_fsi.top -r polyAT_fsi.rst /
-ref polyAT_fsi.crd &
```

This job will take awhile, about 17 minutes of cpu time. You can use the `top` command to follow the progression. Also, **sander** will write a file in the directory you started the job in

called `mdinfo`. This file will be updated regularly. Look for the value of NSTEPS. When it hits 1000, your job is done. You can also use the `.out` file as it, too, is regularly updated. When the job has completed, take a look at the output file referenced above (`polyAT_fsi.out`). You will notice that electrostatic energy (EEL) drops rather rapidly. The former represents bad contacts and the later is optimizing the favorable water-water and water-DNA electrostatic interactions. The RESTRAINT energy represents the effect of the harmonic restraints and it rises rapidly.

To view the final structure run **ambpdb**. The topology file is p`olyAT_fsi.top`, the coordinate file is p`olyAT_fsi.`rst and the pdb file is p`olyAT_fsi_min.`pdb. You can bring up multiple structures in **rasmol**.

Alternatively, you can just use the topology and coordinate files directly by reading them into **moilview**. If you read in `polyAT_fsi.pdb` and `polyAT_fsi_min.pdb`, you can compare the starting and ending structures simultaneously.

In the Questions section you will be asked to prepare a plot of `polyAT_fsi_min.pdb`. It is probably better to generate this plot now. Refer to the section on '**moilview**/Preparing a Plot' section to guide you through this process.

Questions

1) Plots of the following files are to be made and turned in after you have completed this laboratory. The files are:

   A) `nuc_ter.pdb`
   B) `polyAT_vac.pdb`
   C) `polyAT_vac_fixed.pdb`
   D) `polyAT_fsi.pdb`
   E) `polyAT_fsi_min.pdb`

2) Find and open the file `parm94.dat`. Open the file and find the force field parameters for an sp^3 hybridized aliphatic carbon atom for the following:

   A) Atom types, bond length and stretching constant an sp^3 hybridized aliphatic carbon and a carbonyl carbon
   B) Atom types, bond angle and bending constant for an sp^3 hybridized aliphatic carbon and a carbonyl carbon of an amide
   C) Atom types, torsional barrier, and phase angle for an $-CH_2-S-S-CH_2-$

3) Occasionally, after a molecular mechanics calculation has been run to fix hydrogens, molecular mechanics calculations attempted with the output files will fail (energies will just increase and the calculation will 'blow up', usually giving an `LINMIN` error in the output file). This is usually due to failure of the `fixit.in` file to sufficiently minimize, or fix, the hydrogens. What parameters of the `fixit.in` file should be changed to remedy this situation?

4) In addition to sodium ions and water, other ions and other solvents can be added to systems. What other ions and solvents does amber understand? Hint: Take a look at the files xleap loads when it starts.

5) Extra credit: How much difference is there between your final structure (in `polyAT_fsi_min.pdb`) and your original structure (`polyAT_vac.pdb`)? Hint: **moilview** can be used to do this.

Appendices

Appendix A - `fixit.in`

Below is a copy of the `fixit.in` file used to fix-up the hydrogen positions. The file contains settings for most of the available parameters used to control molecular mechanics and molecular dynamics modeling. However, many of the values are the default values and do not require setting.

```
polyA-polyT 10-mer: minimization to fix up hydrogen positions
 &cntrl
  imin   = 1,
  maxcyc = 500,
  ncyc   = 250,
  ntr    = 1,
  ntb    = 0,
  igb    = 0,
 &end
Restrain the DNA heavy atoms I
5000.0
FIND
 *  CT  *  *
 *  C   *  *
 *  CA  *  *
 *  CB  *  *
 *  CK  *  *
 *  CQ  *  *
 *  CM  *  *
 *  OH  *  *
 *  OS  *  *
SEARCH
RES 1  20
END
Restrain the DNA heavy atoms II
5000.0
FIND
 *  O2  *  *
 *  O   *  *
 *  NA  *  *
 *  N*  *  *
 *  N2  *  *
 *  NC  *  *
 *  NB  *  *
 *  P   *  *
SEARCH
RES 1  20
END
END
```