7.36/7.91/BE.490
Homework 3
Revised 4/12/2004
Due April 22 at 1:00 PM

Using Deep View to study protein structure

Computer graphics provides a powerful tool for studying protein structure. You can learn a lot about a protein just by looking at the properties of its structure. For computer visualization of structures in this course, we will use the powerful program Deep View, developed at GlaxoSmithKline R&D in Geneva and hosted by the ExPASy proteomics server. This is available for many different platforms, including Mac, PC, SGI and Linux. Instructions for starting Deepview on MIT Server can be found at http://web.mit.edu/acs/www/simulation.html#Deep (This program was formerly called Swiss PDB Viewer.) Note that this program only runs on Linux machines and not Sun machines. You can also download a version for your own computer, if you wish, from the site http://www.expasy.ch/spdbv/ If you have access to a PC or Mac, this is probably preferable, as the Linux version appears to contain a few bugs (but we describe work-arounds for them below.) If you use Deep View on MIT Server, try to get a fast machine, as it's very computationally demanding. The IBM's are the fastest Linux machines in the clusters. They're in clusters W20-575 at the back, 66-080, and 2-225.

The quickest way to learn to use Deep View is to do the tutorial written by Prof. Gale Rhodes, at the University of Southern Maine. Go to http://www.usm.maine.edu/~rhodes/SPVTut/ and, starting with the Overview section, work your way through the tutorial parts 1 - 6. Referring to the remaining parts of the tutorial may also be helpful for this problem set. Different versions of Deep View may have slightly different user interfaces, so you may occasionally need to click around a bit to find what they're referring to or ask a teaching assistant. Take time as you are doing this to play around with different features of the program. Deep View can be a great resource for you in your research, so this is time well spent.

1. Familiarizing yourself with Deep View

A study by Gidh-Jain et al in 1993 was aimed at determining

how mutations in glucokinase (same as hexokinase IV from problem set 1) altered its function. You can obtain a PDF version of this article for free at http://www.pnas.org/cgi/reprint/90/5/1932. They performed biochemical assays on the wild-type and mutant proteins and calculated their activity. In addition the crystal structure of the related yeast hexokinase B (solved in 1978) was available. The human and yeast sequence are ~30% identical and the authors 'predicted' that the arrangements of a-helices and b-sheets would be similar between the two proteins. They mapped the sites of mutations to the crystal structure and concluded that mutations fell into two groups. One group of mutations was located in the active cleft separating the two domains of the protein and the second group was on the outer side of the protein in a place predicted to undergo a substrate induced conformational change that results in the closure of the active site cleft.

A quick search of the Protein Data Bank for human glucokinases reveals a structure that was added recently (March 30, 2004). Download the structure 1V4S to your computer and open it in SwissPDB viewer. In the toolbar (you can get to the toolbar form the Wind menu) there is an icon showing a page with lines. Clicking on this will open the PDB text file for the structure. Using the text file or the structure viewer answer the following:

a. How was the phase problem solved for this structure? Discuss the pros and cons of this method vs. alternatives.
b. What was the free R value for this structure? What does this value mean? Does it give you confidence in the structure?
c. What was the RMSD of the bond angles and dihedral angles from ideal values? Which one is larger? Is this what you would expect? Why is this an important criterion?
d. List the crystallization conditions for this protein. Is the structure likely to be physiologically relevant?
e. List the order of secondary structural elements from N to C terminus in this structure.
f. List the heteroatom(s) (non-protein atoms), if any, included in this structure.
g. Now go to the Control Panel (you can get to it form the Wind menu). This is a quick way to find residues, change views of the molecule, etc. By default all residues and their side chains are visible (denoted

with a “v”). What are the N- and C-terminal residues that appear on the structure?

Table 1 shows a list of common mutations in glucokinase that lead to non-insulin-dependent (type 2) diabetes mellitus (from Gidh-Jain et al).

Table 1.
Gly175→Arg ^
Val182→Met ^
Val203→Ala *
Thr228→Met *
Glu256→Lys *
Gly261→Arg *
Glu279→Gln *
Gly299→Arg *
Glu300→Lys or Gln *
Leu309→Pro *

* Predicted to be near the active site
^ Predicted to participate in the conformational change that results in cleft closure.

Label and show the sides chains of all of these amino acids. On the control panel window, check the “Show”, “side”(to see the side chain) and “labl”(to show the residue and position) column. Hide the side chains and labels for the rest of the sequence. Hint: To check or uncheck all the residues, place the mouse on the column you want to modify and shift-click. Do this first, then add the amino acids you want to see.

h. Are any of the amino acids that are not adjacent in the primary structure close in space in the tertiary structure? Use the move, center, zoom and rotate tools in the Toolbar menu to look at the structure. Save this structure view for later (Save layer command). Print a copy of this layer and submit it with your homework assignment. (Note: if you have trouble printing, you may need to take a screen shot of your protein and print the image. Black and white print outs are fine.)
i. Do you think the mutations fall into two classes like the authors described? Explain.

Now that both the human and yeast structures are available, you can test if the author’s prediction that the yeast and

human glucokinase structures were similar was correct. Download and open the PDB file 2YHX.

j. On the 'group' column of the control panels many aminoacids are labeled UNK because the authors didn't have the sequence of the protein they were crystallizing. Therefore for some residues, the authors could not unambiguously assign the identity of the residue. For unknown residue 112, list which possible amino acids it could be based on the atoms present.

Superimpose the two structures in order to determine how similar their folds are. To make it easier to see the structures, first display the molecules without side chains (hint: deselect the side chains from the Control Panel and choose 'Show CA trace only' from the Display menu on the tool bar to see the alpha carbons). With the two structures loaded, the Control Panel can now be toggled to display information about each structure. Color the alpha C trace to distinguish the 2 models. Shift-Click on any box in the Col column and choose a color from the color pallet (be sure that backbone + side is selected in the pull down menu to the right of the header in the Col column.) Select Best fit from the Fit menu. Use the alpha carbons for the alignment.

k. How similar are the two protein structures? What can you say about the corresponding location of the residues listed in Table 1 in yeast hexokinase? Do they all lie in regions where both structures are similar?
l. What is the RMSD (considering only alpha carbons) for the superposition? How similar would you say these structures are?
m. Based on your results, do you think the authors 'predictions' were correct? Explain.

2. Homology Modelling

Retrieve a fasta file containing the protein sequence of the protein with accession number JQ2288. This is a transmembrane protein from soybean. It is in general very difficult to obtain structures of membrane proteins. Thus, you would like to construct a model of this protein. Save this file to your home directory.

Start up Deep View without opening a structure file. Choose SwissModel -> Load Raw Sequence to Model and open your fasta file.

In Prefs -> SwissModel, enter your name and email address and click OK.

Choose SwissModel -> Find Appropriate ExPDB templates to find structures that are similar to your sequence of interest. If your Mozilla browser is already started, your window should be redirected to a page that allows you to search for appropriate PDB templates for your sequence of interest. Click on submit to begin finding similar structures.

A list of crystal structures of channels should appear. Download the PDB files of top two BLAST hits and open them in Deep View. The layers window will enable you to switch between the three proteins you now have open.

Superimpose the two structures using Magic Fit. Then select your sequence of interest and repeat a Magic Fit again. Finally choose SwissModel -> Update Threading Display Now. (This may be grayed out if “Update Threading Display Automatically” is set.) You should see your sequence folded up upon your reference PDB structure. To verify that you have done these steps correctly, we suggest that you color the three structures different colors (Color -> Layer) and verify that they are all superimposed on one another.

Choose SwissModel -> Submit Modelling Request. You will be asked for a directory to save your model in. On Linux, the name of this file is by default tmp.html. It will be saved with that filename prefixed by proj_. Mozilla will open at a page that lets you submit your request. Click Browse button, and navigate to where your file was saved. Once the filename has been entered into the corresponding field, click the send request button.

Some time later, you will receive mail from the modeling server. One of the mails will contain your model in the form of a pdb file. Save that file to your home directory. Open your model in Deep View. The three structures will be displayed superimposed. Hide all but your homology model, and turn off the ribbon display.

a. Based on the sequence similarity between your sequence of interest and your two template structures, are these valid templates for homology modeling of your sequence? Explain.
b. Go to Prefs -> Ramachandran Plot and choose to ignore ignore glycines and prolines. View the Ramachandran plot of your homology model (Wind -> Ramachandran Plot). Note that only currently selected amino acids appear in the Ramachandran plot so make sure that you select all amino acids in your model and not any of the other two structures. Are there any residues that fall outside of the "typical" phi/psi space? If there are any, what type of residues are these? Does this analysis increase or decrease your confidence in your model?
c. View the Whatcheck results for your model. Whatcheck is a program that performs various checks on homology models looking for different kinds of errors a PDB file. Give three different problems with the structure of your homology model (other than what was answered in part b) and briefly explain whether you think the error is significant or not.
d. What is the advantage of using more than one template structure during homology modeling? Would it have been worthwhile to include additional template structures from the list of BLAST hits obtained by SwissModel?
e. Do you think that the soybean channel under investigation is actually an aquaporin channel? Discuss.

3. X-ray vs NMR structures

The structure of lac repressor has been solved several times both by X-ray crystallography and by NMR. Open a crystal structure (PDB ID1EFA) and an NMR structure (PDB ID 1L1M) of lac repressor in Deep View. Initially only load one NMR structure. Make sure that you are viewing the Sequences Alignment window and the Layers window.

a. Align the two structures using Magic Fit. What is the all-atom RMSD between the two structures after this fit? Is this value higher or lower than you were expecting? Did the result surprise you? Explain.
b. Provide a possible explanation for why you either do or do not see differences between the two structures. Go beyond merely stating the fact that one structure

was obtained via X-ray crystallography and the other via NMR.

c. Usually, NMR experiments yield a set of solution structures that satisfy the constraints obtained from the data. Now open all 20 structures in 1L1M.Where do you see the most variability between these structures? Why?
d. In what parts of the crystal structure are the B-factors (temperature factors) the highest? Is this the same region where the NMR solutions differ?
e. In some cases with NMR experiments, an averaged energy minimized structure is given rather than a set of structures. Discuss the advantages and disadvantages of the two approaches.
f. Would knowing the NMR structure of a protein aid in solving the crystal structure? Vice versa? Why?

4. Protein side-chain repacking

A particular case of the general protein structure prediction problem is the side-chain repacking problem. In this problem, the backbone and the sequence are fixed and the question is to find the combination of amino acid rotamers (conformational isomers), which makes the lowest energy structure. Repacking is useful in a number of ways. It can be used to model the structural response of a protein to one or more mutations, and it can be used as the final step when building a homology model. Another way in which repacking can be used is to determine the reliability of a particular energy function. If a protein crystal structure is available, one can take its backbone and sequence and repack it given any particular energy function. Comparing the wild type structure to the repacked structure gives one an idea of whether the energy function has the ability to distinguish the native (or near native) conformation as having minimal energy.

Download the file 1FXD.pdb from the class website (not the PDB). Make another copy of it and call it "1FXD_repacked.pdb" (on MIT Server "cp 1FXD.pdb 1FXD_repacked.pdb"). Open this second copy in SwissPDB Viewer (on MIT Server **"spdbv 1FXD_repacked.pdb &"** – will open SwissPDB Viewer in the background). This is the structure of ferredoxin II from *Desulfovibrio gigas*.

a. The sidechain assignment problem is often referred to as a 'combinatorial' problem. Why is this?

b. Let's suppose you have come up with an amazingly fast (and accurate) energy function, which can assign an energy to one combination of rotamers for our structure in 1 CPU cycle. Lets also suppose you have access to an array of 1,000 5 GHz CPUs (which run nothing else but your energy code). Calculate how much time it would take for you to exhaustively enumerate all rotamer combinations for this structure and find the lowest one (assume that each amino acid has 15 rotamer states). Express this in "graduate student work-year equivalents", or some other measure that is meaningful for you.

Go to the Control Panel window and select all amino acids (alternatively, from the Select menu, choose All). Then go Tools Fix Selected Sidechains Quick and Dirty. This function is intended for relieving possible unfavorable steric interactions of the sidechains with the rest of the structure (after a mutation is introduced, for example). This is equivalent to doing a repacking with a very simple energy function (actually, the score or energy each rotamer gets is a combination of the number of clashes it makes with the structure and the number of hydrogen and disulfide bonds it is involved in – for details, see http://www.usm.maine.edu/spdbv/text/mutation.htm). The "quick and dirty" part means that it applies a rapid minimization algorithm, which gives reasonable results but is not guaranteed to give the absolute minimum. Save this structure by going to File Save Layer, navigate into the directory where the original file was, click "Save" and select OK to overwrite the old file (note: do not try to edit the name of the file – this feature is a bit buggy. You'll need this structure for the python problem. Click Display Show H-bonds to hide the hydrogen bonds. Then open the original copy of the file (on MIT Server **"spdbv 1FXD.pdb &"** – this will attach the new file to your previous session). You now have both structure – the original and the repacked one in separate layers. For both layers go to Color By Accessibility (to switch between layers use the Layers Info window; it is located under Window Layers Info).

c. Look at the difference between the wild type and the repacked structures. What regions, or amino acids, are least similar between the two structures? Give some specific residue numbers where the sidechain conformation is wrong.

d. Which region of the molecule is best repacked? (i.e. where is the difference between the two structures smallest?) Give residue numbers for some sites where the side-chain structure is predicted most accurately.
e. Give three reasons why the repacked structure might not agree perfectly with the experimentally determined structure.

5. Analyzing a repacked structure using python

Below is a python program, which reads two PDB files of identical sequence and calculates the room mean square difference (RMSD) between the coordinates of the corresponding sidechains. The names of the two input files should be specified on the command line. It assumes that the two files have exactly the same sequence (the same atoms in the same order).

```
import sys
import math
import re

# Check usage
if (len(sys.argv) < 3):
  print "Error in usage!"
  sys.exit()

# read in PDB files
file1 = sys.argv[1];
file2 = sys.argv[2];
f = open(file1);
lines1 = f.readlines()
f.close()
f = open(file2);
lines2 = f.readlines()
f.close()


# Calculate and print RMSDs
i1 = -1
i2 = -1
pri = "" # previous residue index
prn = "" # previous residue name
sad = [] # array of squares of sidechain atom deviations of the current
residue
while (1):
  f1 = 0
  for i1 in range(i1+1, len(lines1)):
    if not(re.match("^ATOM", lines1[i1])):
      continue
    rn1 = lines1[i1][17:20]
    ri1 = lines1[i1][22:26]
    an1 = lines1[i1][12:16]
    if (re.match("(^\s*H\s*$|^\s*C\s*$|^\s*O\s*$|^\s*N\s*$|^\s*CA\s*$)",
an1)):
      continue
```

```
      f1 = 1
      break

    f2 = 0
    for i2 in range(i2+1, len(lines2)):
      if not(re.match("^ATOM", lines2[i2])):
        continue
      rn2 = lines2[i2][17:20]
      ri2 = lines2[i2][22:26]
      an2 = lines2[i2][12:16]
      if (re.match("(^\s*H|^\s*C\s*$|^\s*O\s*$|^\s*N\s*$|^\s*CA\s*$)", an2)):
        continue
      f2 = 1
      break

    if (f1 and f2 and not((rn1 == rn2) and (an1 == an2) and (ri1 == ri2))):
      print "Error - mismatched lines:\n%s%s" % (lines1[i1], lines2[i2])
      sys.exit()

    # have we started a new residue?
    if (not(ri1 == pri) or (not f1) or (not f2)):
      if (len(sad) > 0):
        rmsd = 0
        for i in range(len(sad)):
          rmsd += sad[i]
        rmsd = math.sqrt(rmsd/len(sad))
        print "%s %s %.2f" % (prn, pri, rmsd)
      # reset all arrays
      sad = []
      prn = rn1
      pri = ri1

    if (not(f1) or not(f2)):
      break

    # now process atom deviation
    x1 = float(lines1[i1][30:38])
    y1 = float(lines1[i1][38:46])
    z1 = float(lines1[i1][46:54])
    x2 = float(lines2[i2][30:38])
    y2 = float(lines2[i2][38:46])
    z2 = float(lines2[i2][46:54])
  sad.append((x1-x2)*(x1-x2) + (y1 - y2)*(y1 - y2) + (z1 - z2)*(z1 - z2))
```

Note that since glycines do not have a side chain (it's a hydrogen, which is usually not explicitly present in PDB files) they are not listed in the output.

a. Apply the provided program to 1FXD.pdb and 1FXD_repacked.pdb. Sort the output values in descending order (you can use Excel or whatever program you want) and look at the top 10 entries. What is common about the corresponding positions? What about the amino acids?
b. Modify the provided python program so that instead of computing the sidechain RMSD between two proteins of identical sequence, it instead computes the c-alpha

RMSD between two user-specified sets of residues in two proteins (possibly of different sequence) and prints the value to the screen.

For instance,

```
> rmsd_backbone.py test1.pdb 20 34 test2.pdb 55 69
```

would give the C-alpha rmsd between residues 20-34 in test1.pdb and residues 55-69 in test2.pdb.

Submit your program on line. Call it rmsd_calpha.py. For this program, you will not be given sample input or output.