CS483 Assignment #2:

Working with Surfaces, Chi1 angles, and Rotamers

Due date: Thursday Feb. 5. "Early Bird" Due date: Tues. Feb. 3 to get 5 bonus marks.

Marks: (The assignment will be marked out of 80).
Ex. 1: [6], **Ex. 2**: [a:6 + b:6 + c:8 + d:4 = 24], **Ex. 3**: [10], **Ex. 4**: [20] , **Ex. 5**: [20]

Exercise 1: Indices and residue positions

Consider the following shell interactions:

```
>>> opkProt = chimera.openModels.open('1opk', type='PDB')[0]
>>> resAtIx0 = opkProt.findResidue(0)
>>> resAtIx0.type
u'ASN'
>>> seqA = opkProt.sequence('A')
>>> seqA.resMap[resAtIx0]
43
>>> resAtIx0.id.position
83
```

This code fetches the PDB file with ID = "1OPK". This protein (c-Abl tyrosine kinase) has been studied because of its relationship to chronic myelogenous leukemia. Unfortunately, the first 82 residues of the protein are missing in the structural X-ray analysis (they are in a long loop that is difficult to localize during crystallization). In the preceding code, we have three numbers: 0, 43, and 83. Explain the significance of each. How do they relate to the sequence display (menu item **Favorites... Sequence**) and the label of the first residue in the display of the protein? Explain the results that you get when the following two expressions are executed in the shell:

```
seqA.residues[43].type
seqA.residues[42].type
```

Exercise 2: Residue burial

Run the following script:

```
import chimera
from chimera import runCommand
pdbIDchars = raw_input("Type in a 4 character PDB ID or a PDB file
pathname: \n")
```

```
prot = chimera.openModels.open(pdbIDchars, type="PDB")[0]
runCommand("surface")
```

If you run the script for a simple protein such as crambin (1CRN) you will see that Chimera puts a surface around the protein structure. More details about the surface command can be found at:

https://www.rbvi.ucsf.edu/chimera/docs/UsersGuide/midas/surface.html

When a surface is generated, Chimera gives each residue object and each atom object additional attributes that describe their interaction with solvent. The attributes are `areaSAS` and `areaSES` where SAS and SES are acronyms for Solvent Accessible Surface and Solvent Excluded Surface. We will focus on `areaSAS` since that will be needed in the assessment of residue burial in Part (b). You will not be finding buried residues using menu invocations as in http://www.cgl.ucsf.edu/chimera/videodoc/surfaceresidues/index.html, but the criterion `r.areaSAS < 1` will be our definition of residue burial for your Python scripts. To begin, do the following exercise:

a) Add statements to the above script so that it iterates through the residue list of the protein and prints out a table with a row for each standard residue. A row should contain the residue position, residue type and the `areaSAS` attribute for the residue. Test your program on the following two proteins: 1CRN and 2QDV.
The second protein might give you some trouble because the PDB file gives two residues at position 28 (MHO is oxymethionine and MET is methionine). But the surface calculation only provides an `areaSAS` for MHO which your script should ignore because it is not a standard amino acid. To get past the `AttributeError`, read about the **`try`** statement in Python. The next exercise will do calculations related to residue burial.

b) For this exercise you will write a script that works through a list of several proteins to gather statistics that will help to assess the likelihood that an amino acid is buried. For each standard residue, we want to calculate the fraction of those residues that have an `areaSAS` that is less than 1 square Angstrom. This fraction is to be expressed as a percentage which you will then compare with the Kyte-Doolittle hydrophobicity value of that residue. The input list can be found in the file for this exercise: PDBselectSingleChainProteinList50chains.txt. If you get Chimera warnings that a surface calculation has failed you can simply ignore it (your **`try`** statement will ensure that you only deal with residues that have an `areaSAS` attribute). After all the proteins are processed your script should print out a table with 20 entries and having the following column headings: `Residue Count`, `Residue Type`, `Percentage Residues Buried`, `Kyte-Doolittle Hydrophobicity`. For neatness, the headings should span three lines. Residue count is the total number of residues processed for that type. For floating point numbers use 4 decimal places (format `%7.4f`). Hand in a listing of the results so that the TA does not have to verify your script by running it against the entire data set.

c) Add more code to your script so that it generates a scatter plot for the output that was generated in the previous exercise. The scatter plot will have 20 points (also called "markers"). The x-axis should correspond to Kyte-Doolittle Hydrophobicity and the y-axis should correspond to the Percentage Buried values. Scatter plot generation is discussed on page 370 of the text and the code can be found at the structuralbioinformatics.com website. Be sure to use appropriate titles and axis labels. Change the plot settings so that the data is presented with reasonable visual clarity. For example, the `aspect` parameter in the `add_subplot` function should be set to "`auto`". We also want the markers in the plot to give a visual indication about the nature of the residue. While going through the 20 residue types, the `scatter` function should use markers that change according to the category of amino acid (as described on pages 7 and 8 of the text). The following rules for marker colour and marker shape should be used:

Hydrophobic residues	colour: "orange"	marker shape: "o" (circle)
Polar uncharged residues	colour: "blue"	marker shape: "d" (diamond)
Polar charged (+) residues	colour: "blue"	marker shape "^" (up triangle)
Polar charged (−) residues	colour: "blue"	marker shape "v" (down triangle)

If all goes well you should notice various tendencies: The hydrophobic residues will tend to have a higher Percentage Buried value (i.e. they are more likely to be buried in the protein core) while the polar residues, especially the charged residues, will tend to have a lower Percentage Buried value (i.e. they are more likely to be in contact with the water solvent). The script will generate a .png file for the output and this should be submitted along with the script.

d) Run Script D_15 Pie plot & GUI.py which can be found on the Structuralbioinformatics.com website. When the GUI asks for data you may use a text file containing sample data such as:

```
Foo values for some animals
323.4 cat
448.2 dog
553.5 rat
116.2 elk
233.1 bat
```

The script will generate a pie chart with an appearance similar to that shown in Figure B.11 (colour pages in the text). Run the script again using data extracted from the processing done in Exercise 2(c) above. In other words, the first line for the .txt file is just some descriptive text followed by 20 lines each line corresponding to one of the standard residue types. Each line contains the number of buried residues for the residue type followed by a three letter code to identify the residue type. Numbers do not have to add up to 100 (the script will calculate percentages). The idea of the pie chart is to visually show the typical constituents of the "average" set of buried residues.

Exercise 3: Computing changes in dihedral angles

Write a function that accepts three arguments: `r_a`, `r_b` and `threshold`. Both `r_a` and `r_b` are residue objects and `threshold` is a floating point number between 0.0 and 180.0. For our purposes, the value of `threshold` will usually be 40.0. The function should return a 2-tuple containing two Boolean values. The first Boolean entry will be `True` if and only if the separation between the dihedral angles `r_a.chi1` and `r_b.chi1` is less than or equal to the `threshold` value. The second Boolean entry will be `True` if and only if the first Boolean entry is `True` *and* the separation between the dihedral angles `r_a.chi2` and `r_b.chi2` is less than or equal to the `threshold` value. Note: you cannot simply calculate the absolute value of `r_a.chi1 - r_b.chi1` followed by a comparison with the `threshold` value. This will work for some pairs of angles (for example, two small angles such as 44.0 and 67.0 have a separation of 23.0 which is less than the threshold value of 40.0) but will fail for two dihedral angles such as 170.0 and −175.0 which have a separation of only 15 degrees. Remember: dihedral angles are assumed to be in the range (−180 to +180) and the maximum separation will be 180.0. If the residues do not have `Chi1` dihedral angles then the function should return (`None`, `None`). If the residues have a `Chi1` dihedral angle but not a `Chi2` dihedral angle then the return value will be either (`True`, `None`) or (`False`, `None`) depending on the separation between the `Chi1` dihedral angles.

To test your function write a mainline program that does the following:

i. Get a PDB ID from the user.
ii. Fetch the protein and replace all the standard residues (except for GLY and ALA) with the rotamer having the highest probability (see code below). You should also avoid the very first residue because we only want to deal with rotamers that are backbone dependent.
iii. Fetch the *same* protein again. If your script executes `runCommand("ribbon; repr stick; show")` the display should show both models with the sidechains protruding from the ribbons. The alpha carbons of a residue and its rotamer will overlap in 3-space but the rest of the side chain will likely be different.
iv. Print out a four column table with the following entries in each row: residue position, residue type, and the two entries in the tuple returned by the function that you are testing.
v. At the very end of the print out provide summary information that gives the percentage of rotamer `chi1` angles that are within 40 degrees of the native (PDB) value and the percentage of rotamers with both the `chi1` and `chi2` angles within 40 degrees of the native values.

To replace a residue `r` of the first protein model with the highest probability rotamer use the following code in your loop:

```
try:
    (flag, rots_L) = getRotamers(r)
    useRotamer(r, [rots_L[0]])  #Index 0 gets most probable rotamer
except NoResidueRotamersError:
    continue
except LimitationError:
    continue
```

This code will require the following imports:

```
import Rotamers
from Rotamers import getRotamers, useRotamer, NoResidueRotamersError, LimitationError
```

Test the code on a simple protein such as Crambin (1CRN) so that you can easily see the results of the rotameric substitutions. In particular, when your function returns a tuple such as (`False`, `False`) for some backbone position, the wide separation between the rotamer and its native residue should be easily seen at that backbone position in the display.

Exercise 4: BBDep prediction and side chain burial

Consider the table shown on the webpage at: http://dunbrack.fccc.edu/scwrl4/. In this exercise we are interested in the 18 Back-Bone Dependent (BBDep) values in the columns dealing with `Chi1` prediction accuracy. As described on the webpage, this column is simply reporting the prediction accuracy that one should expect if the most probable rotamer is used without any extra computation being done to minimize some energy function (which would also avoid steric collisions). The idea in this exercise and the next exercise is to see if we can establish a correlation between the values in the column and some other feature of a residue (for example side chain burial or hydrogen bonding of the side chain).

If you study the BBDep column you will notice that some of the hydrophobic residue types are associated with a higher prediction accuracy. So, we pose the question: Is the use of the most probable rotamer consistent with more prediction success if the residue is buried (in contrast to a residue that is exposed to solvent)? In other words, we want to redo the calculations that produced the 18 numbers in the BBDep column but instead of 18 categories (corresponding to residue types) we want to use 36 categories corresponding to residue types across both classifications: buried and exposed.

The script for this exercise should do the following:

i. Use `tkFileDialog` to get a file name from the user. As done in Ex 4 of Assignment 1 we will be processing a subset from PDBselect and the list of PDB identifiers will be in the text file prepared by the user.
ii. For each protein in the list, we fetch the protein and immediately put a surface around it so that the `areaSAS` values are created for the residues in that model. This is followed by a fetch of the same protein and the setting of each standard residue to the rotamer with the highest probability (as before, skipping GLY and ALA).
iii. The function that you scripted in the previous exercise should be used to evaluate whether the rotamer has a `Chi1` value that is within 40 degrees of the true value as seen in the second protein. You can ignore the `Chi2` dihedral angles. As you process the residue/rotamer pairs, you should gather data that allows you to get the following information *for each residue type*: the total number of residues exposed, the total number

of residues buried, the number of exposed residues with the rotamer having a `chi1` angle within 40 degrees of true, and the number of buried residues with the rotamer having a `chi1` angle within 40 degrees of true, These totals are accumulated across all proteins in the list provided by the user so that after all the proteins have been processed you will have 4 totals for each of the 18 different residue types.

iv. Report the results by printing out a table, one row for each of the 18 residue types. Each row contains the following information: residue type, total number of exposed residues, the number of these rotamers with `Chi1` values that are within 40 degrees of true (expressed as a percentage as seen in the Dunbrack webpage), total number of buried residues, the number of these rotamers with `Chi1` values that are within 40 degrees of true (also expressed as a percentage), the total number of residues (sum of the exposed and buried residue counts), and the number of these rotamers with `Chi1` values that are within 40 degrees of true (also expressed as a percentage). You should expect that this last column will be very similar to the BBDep column seen on the Dunbrack Lab webpage. A final summary line should give the same values but with the total counts applying to all residues i.e. going across all the 18 residue types.

In your estimation, does buried vs. exposed make any significant difference to the accuracy of the prediction?

Exercise 5: BBDep prediction and hydrogen bonding in the side chains

Redo the previous exercise but modify the script so that we again have 36 categories. This time instead of working with exposed and buried classifications for each residue type we want to establish whether or not the side chain has atoms that act as hydrogen bond donors or acceptors. To decide whether a side chain is involved with hydrogen bonding use the same criteria that you employed in Exercise 5(b) of Assignment 1. The results should be presented as table just as done in Ex. 4 above but with column headings that relate to the presence of hydrogen bonding instead of residue burial. Caution: for each PDB entry in the given list, do your hydrogen bond analysis before the second protein is fetched. In particular, you do not want to have hydrogen bonds created between the two models!

The results of this exercise will be more interesting. You should provide a short paragraph that attempts to explain the significance of the result.