Workshop #6: Packing & Design

Rosetta uses a Monte Carlo optimization routine to pack side chains using a library of conformations, or rotamers. This operation can be used for side-chain packing for operations like refinement or for designing optimal sequences. This workshop will examine both capabilities.

Suggested readings

1. J. Desmet *et al.*, "The dead-end elimination theorem and its use in protein side-chain positioning," *Nature* **356**, 539-543 (1992).
2. B. Kuhlman & D. Baker, "Native protein structures are close to optimal for their structures," *PNAS* 97, 10383, 2000.

Side Chain Conformations, the Rotamer Library, and Dunbrack Energies

Begin by loading cetuximab from PDB 1YY8.

1. What are the φ, ψ, and χ angles of residue K49?

   φ: ________ ψ: ________ χ_1: ________ χ_2: ________ χ_3: ________ χ_4: ________

2. Open `asp.bbdep.rotamers.lib` by unpacking `asp.bbdep.rotamers.gz` from the directory `rosetta_database/rotamer/ExtendedOpt1-5`. Find the φ/ψ bin for lysine at residue 49 and find the nearest rotamer. What are the χ angles and standard deviations of this rotamer? What is its probability?

   χ_1: ________±______ χ_2: ________±______ χ_3: ________±______ χ_4: ________±______

   P = ___________

3. Score your pose with the standard full-atom score function. What is the energy of K49? Note the Dunbrack energy component (`fa_dun`), which represents the side-chain conformational probability. Does it match what you found in the table? (You will need to convert between probability and energy; use $kT = 1$.) If not, why not?

4. Use `pose.set_chi(i, res_num, chi)` to set the side chain of residue 49 to the all-anti conformation. (Here, `i` is the χ index, and `chi` is the new torsion angle in degrees.) Re-score the pose and note the Dunbrack energy. Does it match the probability in the table? _____ Is this conformation valid for cetuximab (*i.e.*, is the total score of this residue acceptable)? _____

Monte Carlo Side-Chain Packing

Side-chain packing can be done in a Monte Carlo search routine that iteratively swaps rotamers of a random residue and tests each move using the Metropolis criterion. Rosetta has such a routine pre-packaged as a `Mover` that carries out a simulated annealing search each time it is applied. The specific scope of the packing is specified in a `PackerTask` object, which is similar to a `MoveMap` in that it specifies degrees of freedom. We can specify via commands or from an input file our settings for a `PackerTask`. Create a PackerTask as follows. This will set the task to allow packing only of residue 49:

```
task_pack = standard_packer_task(pose)
task_pack.restrict_to_repacking()
task_pack.temporarily_fix_everything()
task_pack.temporarily_set_pack_residue(49, True)
```

The default task allows *any* amino acid residue to be swapped in for another; that is, it would simulate a protein variant as a result of mutation. This would be useful for protein design but not for side-chain packing. `restrict_to_repacking()`only allows rotamers from the current residue at that position to be used. We can confirm our settings using

```
print task_pack
```

We now can construct a `PackRotamersMover`:

```
pack_mover = PackRotamersMover(scorefxn, task_pack)
```

5. Apply the `PackMover` above to your pose with `pack_mover.apply(pose)`. Now what are the χ angles of K49? Which rotamer is this? What is the Dunbrack energy?

6. What is the new total energy of K49? _________ Why did Rosetta pick this rotamer? Answer this in terms of the components of the score function and in terms of the residues with which K49 interacts.

Packing for Refinement

Side-chain packing can be used when converting a pose from centroid to full-atom mode, and it is used extensively in full-atom refinement calculations. Let’s examine how packing improves scores.

Use your code from Workshop #5 to create a centroid-representation model for RecA protein domain 2. Save that centroid "decoy" so that we can compare several basic refinement steps.

7. Load the centroid decoy and convert it to full-atom representation using the `SwitchResidueTypeSetMover`. Save this starting configuration for future use. Score the pose with the standard score function. Why is the score so high?

8. Create a default `PackRotamersMover` with a `PackerTask` that allows all residues to vary χ angles. Create a test pose from your start pose and pack the side chains. What is the new pose score? _________

9. Reset the test pose to the start configuration. Create a `MinMover` using the Davidson-Fletcher-Powell minimization scheme by applying the method `min_type("dfpmin")` to your mover. Create a `MoveMap` that allows χ angles but *not* φ/ψ/ω angles to vary. Apply the `MinMover` and rescore the pose. How does this energy compare?

10. Again, reset the test pose. Apply the packer and then minimize on the χ angles. Now what is the final score? _________

For fun, you might examine the individual residue energies to find the residues most responsible for the score changes. Typically, a small number of residues may make clashes that can be resolved using the χ angle minimization, which allows off-rotamer side-chain conformations.

Design

Design calculations can be accomplished simply by packing side chains with a rotamer set that includes all amino acid types. That is, when the Monte Carlo routine swaps rotamers, it could replace the existing side chain with another conformation of the same residue or some conformation of a different residue type. Trial mutations are accepted or rejected with the Metropolis criterion, and the standard full-atom energy function is supplemented by a reference energy term, `ref`, which represents the relative energies of each residue type in an unfolded peptide.

Design operations are easiest to specify through a data file called a "resfile." You can create a resfile for a given pdb file or pose using the following `toolbox` methods:

```
from toolbox import generate_resfile_from_pdb
generate_resfile_from_pdb("1YY8.pdb", "1YY8.resfile")
from toolbox import generate_resfile_from_pose
generate_resfile_from_pose(pose, "1YY8.resfile")
```

Inside the resfile you will see a list of all residues and their chain with `NATRO` next to that, indicating that the position is set to use the <u>nat</u>ive <u>ro</u>tamer. `NATRO` can be changed to any of the following with a text editor:

`NATRO`	use native amino acid and native rotamer (does not repack)
`NATAA`	use native amino acid but allow repacking to other rotamers
`PIKAA ILV`	use only the following amino acids and allow repacking between them
`ALLAA`	use all amino acids and all repacking

Edit the resfile to force residue 49 to be glutamic acid (`49 A PIKAA E`) and save the file as `1YY8-K49E.resfile`. Create a new task for design from the resfile:

```
task_design = TaskFactory.create_packer_task(pose)
parse_resfile(pose, task_design, "1YY8-K49E.resfile")
```

11. Score the original conformation from the pdb for reference. Create a new `PackResiduesMover` with the design task and use it to mutate residue 49 to glutamic acid. What is the predicted Δ*G* of the mutation? _______ Is this a stabilizing mutation? _____

12. Note the residue reference energy term (`ref`) in the scoring function. What is this value before and after you mutated the residue? What does this energy represent?

13. Create a new `PackerTask` and `PackRotamersMover` using a new resfile that allows residue 49 to be designed freely (`49 A ALLAA`), and apply the mover. What residue does Rosetta choose? _____ Why?

14. Create your own resfile that will restrict residue 49 to only negatively charged residues using the resfile line `49 A PIKAA DE` and re-apply the design mover. Now what residue is chosen? _____ What is the new residue energy, and why (physically) is it less favorable than the last design?

15. Let's try to make this design more favorable. Select several surrounding residues for design, and set them also to enable mutations to any residue. Call the design mover again. Now what do you find?

It should be noted that PyRosetta includes a handy `toolbox` method `mutate_residue()` that will change a specified residue in a given pose into another. However, the rotamer of this new residue will not be optimized. For example:

```
from toolbox import mutate_residue
mutate_residue(pose, 49, 'E')
```

Programming Exercises

1. *Refinement and discrimination*. Download the "single misfold" decoy set from the Decoys 'R Us repository at dd.compbio.washington.edu/ddownload.cgi?misfold. (Documentation for this project is at dd.compbio.washington.edu.) This repository has a single "correct" and "incorrect" predicted structure for several proteins. For this exercise, analyze pdbs 2CI2 and 2CRO; each has two "incorrect" structures offered. (Technical note: These decoys have an empty occupancy field in the PDB `ATOM` records; a value of 1 needs to be added before Rosetta will load these structures.)

   Write a program that will calculate and output the score for each decoy (i) as is from the PDB file, (ii) after packing only, (iii) after minimization only, and (iv) after packing and minimizing. For each of the four cases, compare the scores of the "correct" structure with those of the "incorrect" structure. Which schemes successfully discriminate the correct structures?

2. Write a refinement protocol that will iterate between side-chain packing, small and shear moves, and minimization. Where is the best place to position the Monte Carlo acceptance test? Test your protocol by making 10 independently-refined structures for the correct and incorrect decoys of 2CRO from the Decoys 'R Us single misfold set. Is this protocol able to discriminate the correct decoy? Submit your code.

3. HIV-1 protease is a major drug target for antiretroviral therapies. Protease inhibitors are designed from substrate peptide mimics. We will attempt to take a natural substrate peptide of HIV-1 protease and design it for improved binding — potentially to serve as a good template for drug design. Use PDB file 1KJG for the following analysis.

   a. Turn on side-chain packing for the protease active site (residues 8, 23, 25, 29, 30, 32, 45, 47, 50, 53, 82, and 84 of both chains A and B) and for the peptide (residues 2–9 on chain P; all of these numbers follow the PDB numbering).

   b. Repack the above side chains and then energy minimize those same side chains with the backbone fixed. Generate 10 decoys and record the energies for each decoy. This will represent the reference state: the wild-type peptide bound to the protease.

   c. For residue 2 of the peptide (chain P), allow repacking to any of the 20 amino acid residues, while leaving the packing and side-chain minimization the same as in step b. Generate 10 decoys and record the energies. These will represent single mutants at that residue position.

   d. Repeat step c for each of the other 8 residues in the substrate peptide.

   e. Take the lowest energy for each mutation position and compare it to the lowest energy for the wild type. Do single mutants at any of these positions improve the energy over the wild type? Which ones? By how much? Which energy components are mostly responsible?

   f. Which peptide residue positions are easiest to improve? Which positions are the hardest?

   g. Are there any other trends? Hydrophobic *vs.* polar, bulky residues *vs.* small residues, *etc.*?

   h. Altman *et al.* (*Proteins* 2008) found, using their own computational design algorithm, that the most favorable sequences were a triple mutant E3D/T4I/V6L, a single mutant T4V, and a single mutant E3Q. How do their results compare with yours?

   i. Natural substrates are often sub-optimal binders. Why would this be advantageous?

4. *Effect of backbone conformation on design*. HIV-1 protease is promiscuous, meaning it can cleave a wide range of peptides beyond the ten natural substrates of the virus. Let's examine the preferences of the enzyme through Rosetta design calculations.

   a. Download HIV-1 protease in complex with CA-P2 peptide (1F7A). Select the eight peptide residues for unrestricted design and let Rosetta redesign the substrate sequence. What is the new sequence and how does it compare to the original? What percent of the original sequence was optimal for its structure?

   b. Download HIV-1 protease in complex with RT-RH peptide (1KJG). (Note that the enzyme is the same here, but it is crystallized with a different substrate.) Again, design the eight substrate residues with Rosetta. What percent of this substrate sequence is optimal for this crystal structure? ____%

   c. How do the designed sequences of (a) and (b) compare? Why should they be the same? Why would they not be the same? What are the implications for the field of computational protein design?

5. Write a program which iterates between design of all residues of a protein and refinement via small, shear, and minimization moves.

Thought Question

1. What is the thermodynamic meaning of the `ref` energy term, and what does it correspond to *physically*?

2. During evolution, the genome sequence may mutate to cause protein sequence changes. Alternately, one could consider the difference in evolutionary propensities for each residue type. How could you derive reference energies from sequence data, and what would that mean?

3. How do Kuhlman & Baker fit the reference energies in their 2000 *PNAS* paper?

References

1. S. C. Lovell *et al.*, "The penultimate rotamer library," *Proteins* **40**, 389-408 (2000).
2. R. L. Dunbrack & F. E. Cohen, "Bayesian statistical analysis of protein side-chain rotamer preferences," *Protein Sci.* **6**, 1661-1681 (1997)