Quick-start guide to protein design using Rosetta 3.1

This is one of a series of tutorials designed to get you started with using Rosetta 3.1. It was produced to accompany Kaufmann et. al. (2010) Biochemistry, and the latest version can be found at http://meilerlab.org/. The purpose of Rosetta's fixed-backone design mode is to predict the optimal sequence that will adopt a given structural fold- the 'inverse folding problem'. The algorithm is described in Kuhlman et. al (2003) Science 302, 1364-1368. It is assumed that you have installed the Rosetta suite from rosettacommons.org, and that you are comfortable working in Linux. You will also want to become familiar with the documentation that can be found in the /manual/ and /demos/ subdirectories, as well as the online Rosetta 3 User Manual, FAQ, and forums at http://www.rosettacommons.org/tiki/.

Setting up the design run

1. You need a protein structure to redesign. There is a sample PDB file provided with this tutorial in the /design/ directory included with this tutorial: 1ubi.pdb, which has been simplified as much as possible using perl scripts found in the /BioTools/ directory of the Rosetta installation. This file has no header, only a single protein chain, and contains no solvent or heteroatoms.
2. Create a flags file to specify the parameters Rosetta will use during this run, including the location and format of input and output files, and options that specify details of the algorithm. There is a sample flags file in the /design/ directory. It is worth reading the file in a text editor becuase it is well-commented and contains pointers to more documentation. The parameters of the run can be changed by editing this file.
3. You need a resfile to specify which residues of the input file will be varied, and what variations are allowed at each position. For example, only rotameric changes could be permitted, resulting in a simple repack of the protein. Or specific replacements can be allowed at specific positions in order to simulate known mutants. Or all residues could be allowed to mutate freely, in order to find the most stable sequence consistent with the given backbone structure. An example resfile is provided in the /design/ directory.
4. Conduct the design run:

```
fixbb.linuxgccrelease @flags > design.log
```

Analysis and notes

1. Examine the output files generated during this design run. This includes the logfile, the pdb structure files (or alternatively, a silent.out file containing multiple structures) and the scorefile. These are all plain text files, so you can read them with a pager or text editor.
2. Rosetta's output structures are often referred to as decoys. Load some of the decoys into your favorite molecular visualization program (Chimera, PyMOL, etc) and compare them with the native ubiquitin structure (1ubi.pdb). Can you find the changes? Do they seem reasonable to you?
3. The output structures can be ranked by the Rosetta energy score, where the lowest energy indicates the best structure. The Rosetta scores are located in the scorefile, but can also be found in the silent.out file, and per-residue scores are at the end of the output PDB files. Do you believe that the redesigned proteins will have a lower energy than native ubiquitin?
4. Repeat the run using the alternative flags file, which uses the second resfile to make targeted mutations in the core of the protein.

```
fixbb.linuxgccrelease @flags2 > design2.log
```